# Table of Contents

- What is Amazon SES Classic ................................................................. 1
  - Features of Amazon SES ................................................................ 1
  - Why use Amazon SES? .................................................................. 1
  - Amazon SES and other AWS services ........................................ 1
  - In this guide .................................................................................... 2

- Quick start .......................................................................................... 3
  - Step 1: Sign up for AWS ............................................................... 3
  - Step 2: Verify your email address ................................................ 3
  - Step 3: Send your first email ....................................................... 3
  - Step 4: Consider how you will handle bounces and complaints .... 3
  - Step 5: Move out of the Amazon SES sandbox ......................... 3
  - Next steps ...................................................................................... 4

- Sending email ..................................................................................... 5
  - How do I send emails using Amazon SES? ................................. 5
  - How do I start? ............................................................................. 6

- Concepts ............................................................................................. 7
  - Amazon SES and deliverability .................................................... 7
  - Email-sending process ................................................................. 11
  - Email format and Amazon SES ................................................... 14
  - Types of credentials .................................................................... 16

- Getting started .................................................................................. 18
  - Using the Amazon SES console .................................................. 18
  - Using Simple Mail Transfer Protocol (SMTP) ......................... 18
  - Using an AWS SDK ..................................................................... 19
  - Before You Begin ....................................................................... 19
  - Send an Email Using the Console ............................................. 19
  - Send an email using SMTP ......................................................... 20
  - Send an email using an AWS SDK ............................................ 30
  - Migrate to Amazon SES from another solution .................... 44

- Setting up email ................................................................................ 46
  - Signing up for AWS ..................................................................... 46
  - Verifying identities ...................................................................... 47
  - Getting your AWS access keys ................................................... 65
  - Downloading an AWS SDK ....................................................... 65
  - Setting up a custom MAIL FROM domain ............................... 65
  - Setting up SPF records ............................................................... 71
  - Getting your SMTP credentials .................................................. 71
  - Moving out of the sandbox ........................................................ 71
  - Configuring custom open and click domains ......................... 77
  - Setting up VPC endpoints ........................................................... 81

- Email sending methods ................................................................. 85
  - Using the SMTP interface ............................................................ 85
  - Using the API .............................................................................. 109
  - Sending personalized email ....................................................... 117

- Email authentication methods ..................................................... 129
  - Authenticating Email with SPF ................................................ 129
  - Authenticating Email with DKIM .............................................. 130
  - Complying with DMARC ............................................................ 142

- Managing sending quotas ............................................................ 144
  - Monitoring your sending quotas .............................................. 145
  - Increasing your sending quotas .............................................. 146
  - Sending quota errors .............................................................. 148

- Using sending authorization ......................................................... 149
  - Overview of Sending Authorization ........................................ 150
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitoring your sender reputation</td>
<td>369</td>
</tr>
<tr>
<td>Monitoring your sending activity</td>
<td>259</td>
</tr>
<tr>
<td>Using configuration sets</td>
<td>249</td>
</tr>
<tr>
<td>Using the reputation dashboard</td>
<td>369</td>
</tr>
<tr>
<td>Monitoring sending using event publishing</td>
<td>288</td>
</tr>
<tr>
<td>Event publishing terminology</td>
<td>289</td>
</tr>
<tr>
<td>Setting up email receiving</td>
<td>231</td>
</tr>
<tr>
<td>Using notifications</td>
<td>238</td>
</tr>
<tr>
<td>Monitor using the API</td>
<td>261</td>
</tr>
<tr>
<td>Calling the GetSendStatistics API operation using the AWS CLI</td>
<td>262</td>
</tr>
<tr>
<td>Calling the GetSendStatistics operation programmatically</td>
<td>262</td>
</tr>
<tr>
<td>Monitoring using notifications</td>
<td>265</td>
</tr>
<tr>
<td>Important considerations</td>
<td>265</td>
</tr>
<tr>
<td>Notifications through email</td>
<td>265</td>
</tr>
<tr>
<td>Amazon SNS notifications</td>
<td>268</td>
</tr>
<tr>
<td>Monitoring sending using event publishing</td>
<td>288</td>
</tr>
<tr>
<td>How event publishing works</td>
<td>288</td>
</tr>
<tr>
<td>How to use event publishing</td>
<td>289</td>
</tr>
<tr>
<td>Event publishing terminology</td>
<td>289</td>
</tr>
<tr>
<td>Setting up event publishing</td>
<td>290</td>
</tr>
<tr>
<td>Working with event data</td>
<td>299</td>
</tr>
<tr>
<td>Tutorials</td>
<td>341</td>
</tr>
<tr>
<td>Viewing the details of a configuration set</td>
<td>251</td>
</tr>
<tr>
<td>Deleting a configuration set</td>
<td>251</td>
</tr>
<tr>
<td>Managing default configuration sets</td>
<td>252</td>
</tr>
<tr>
<td>Managing event destinations</td>
<td>254</td>
</tr>
<tr>
<td>Updating an event destination</td>
<td>255</td>
</tr>
<tr>
<td>Deleting an event destination</td>
<td>255</td>
</tr>
<tr>
<td>Enabling or disabling an event destination</td>
<td>255</td>
</tr>
<tr>
<td>Managing IP pools</td>
<td>256</td>
</tr>
<tr>
<td>Assigning an IP pool to a configuration set</td>
<td>256</td>
</tr>
<tr>
<td>Modifying IP pool assignments</td>
<td>256</td>
</tr>
<tr>
<td>Specifying configuration sets in email</td>
<td>257</td>
</tr>
<tr>
<td>Exporting reputation metrics</td>
<td>257</td>
</tr>
<tr>
<td>Enabling the exporting of reputation metrics for a configuration set</td>
<td>257</td>
</tr>
<tr>
<td>Disabling the exporting of reputation metrics for a configuration set</td>
<td>258</td>
</tr>
<tr>
<td>Monitoring your sending activity</td>
<td>259</td>
</tr>
<tr>
<td>Monitoring using the console</td>
<td>261</td>
</tr>
<tr>
<td>Monitor using the API</td>
<td>261</td>
</tr>
<tr>
<td>Calling the GetSendStatistics API operation using the AWS CLI</td>
<td>262</td>
</tr>
<tr>
<td>Calling the GetSendStatistics operation programmatically</td>
<td>262</td>
</tr>
<tr>
<td>Monitoring using notifications</td>
<td>265</td>
</tr>
<tr>
<td>Important considerations</td>
<td>265</td>
</tr>
<tr>
<td>Notifications through email</td>
<td>265</td>
</tr>
<tr>
<td>Amazon SNS notifications</td>
<td>268</td>
</tr>
<tr>
<td>Monitoring sending using event publishing</td>
<td>288</td>
</tr>
<tr>
<td>How event publishing works</td>
<td>288</td>
</tr>
<tr>
<td>How to use event publishing</td>
<td>289</td>
</tr>
<tr>
<td>Event publishing terminology</td>
<td>289</td>
</tr>
<tr>
<td>Setting up event publishing</td>
<td>290</td>
</tr>
<tr>
<td>Working with event data</td>
<td>299</td>
</tr>
<tr>
<td>Tutorials</td>
<td>341</td>
</tr>
<tr>
<td>Viewing error metrics</td>
<td>237</td>
</tr>
<tr>
<td>Managing IP address filters</td>
<td>236</td>
</tr>
<tr>
<td>Managing receipt rules</td>
<td>234</td>
</tr>
<tr>
<td>Managing receipt rule sets</td>
<td>232</td>
</tr>
<tr>
<td>Managing receipt rule sets</td>
<td>232</td>
</tr>
<tr>
<td>Viewing the details of a receipt rule set</td>
<td>231</td>
</tr>
<tr>
<td>Deleting a receipt rule set</td>
<td>231</td>
</tr>
<tr>
<td>Creating receipt rules set</td>
<td>219</td>
</tr>
<tr>
<td>Creating receipt rule sets</td>
<td>219</td>
</tr>
<tr>
<td>Creating a receipt rule set</td>
<td>219</td>
</tr>
<tr>
<td>Creating a configuration set</td>
<td>249</td>
</tr>
<tr>
<td>Creating configuration sets</td>
<td>249</td>
</tr>
<tr>
<td>Creating a configuration set</td>
<td>249</td>
</tr>
<tr>
<td>Creating configuration sets</td>
<td>249</td>
</tr>
<tr>
<td>Viewing a list of your configuration sets</td>
<td>250</td>
</tr>
<tr>
<td>Viewing the details of a configuration set</td>
<td>251</td>
</tr>
<tr>
<td>Deleting a configuration set</td>
<td>251</td>
</tr>
<tr>
<td>Managing default configuration sets</td>
<td>252</td>
</tr>
<tr>
<td>Managing event destinations</td>
<td>254</td>
</tr>
<tr>
<td>Updating an event destination</td>
<td>255</td>
</tr>
<tr>
<td>Deleting an event destination</td>
<td>255</td>
</tr>
<tr>
<td>Enabling or disabling an event destination</td>
<td>255</td>
</tr>
<tr>
<td>Managing IP pools</td>
<td>256</td>
</tr>
<tr>
<td>Assigning an IP pool to a configuration set</td>
<td>256</td>
</tr>
<tr>
<td>Modifying IP pool assignments</td>
<td>256</td>
</tr>
<tr>
<td>Specifying configuration sets in email</td>
<td>257</td>
</tr>
<tr>
<td>Exporting reputation metrics</td>
<td>257</td>
</tr>
<tr>
<td>Enabling the exporting of reputation metrics for a configuration set</td>
<td>257</td>
</tr>
<tr>
<td>Disabling the exporting of reputation metrics for a configuration set</td>
<td>258</td>
</tr>
<tr>
<td>Monitoring your sending activity</td>
<td>259</td>
</tr>
<tr>
<td>Monitoring using the console</td>
<td>261</td>
</tr>
<tr>
<td>Monitor using the API</td>
<td>261</td>
</tr>
<tr>
<td>Calling the GetSendStatistics API operation using the AWS CLI</td>
<td>262</td>
</tr>
<tr>
<td>Calling the GetSendStatistics operation programmatically</td>
<td>262</td>
</tr>
<tr>
<td>Monitoring using notifications</td>
<td>265</td>
</tr>
<tr>
<td>Important considerations</td>
<td>265</td>
</tr>
<tr>
<td>Notifications through email</td>
<td>265</td>
</tr>
<tr>
<td>Amazon SNS notifications</td>
<td>268</td>
</tr>
<tr>
<td>Monitoring sending using event publishing</td>
<td>288</td>
</tr>
<tr>
<td>How event publishing works</td>
<td>288</td>
</tr>
<tr>
<td>How to use event publishing</td>
<td>289</td>
</tr>
<tr>
<td>Event publishing terminology</td>
<td>289</td>
</tr>
<tr>
<td>Setting up event publishing</td>
<td>290</td>
</tr>
<tr>
<td>Working with event data</td>
<td>299</td>
</tr>
<tr>
<td>Tutorials</td>
<td>341</td>
</tr>
<tr>
<td>Viewing error metrics</td>
<td>237</td>
</tr>
<tr>
<td>Managing IP address filters</td>
<td>236</td>
</tr>
<tr>
<td>Managing receipt rules</td>
<td>234</td>
</tr>
<tr>
<td>Managing receipt rule sets</td>
<td>232</td>
</tr>
<tr>
<td>Viewing the details of a receipt rule set</td>
<td>231</td>
</tr>
<tr>
<td>Deleting a receipt rule set</td>
<td>231</td>
</tr>
<tr>
<td>Creating receipt rules set</td>
<td>219</td>
</tr>
<tr>
<td>Creating receipt rule sets</td>
<td>219</td>
</tr>
<tr>
<td>Creating a receipt rule set</td>
<td>219</td>
</tr>
<tr>
<td>Creating a configuration set</td>
<td>249</td>
</tr>
<tr>
<td>Creating configuration sets</td>
<td>249</td>
</tr>
<tr>
<td>Creating a configuration set</td>
<td>249</td>
</tr>
<tr>
<td>Creating configuration sets</td>
<td>249</td>
</tr>
<tr>
<td>Viewing a list of your configuration sets</td>
<td>250</td>
</tr>
<tr>
<td>Viewing the details of a configuration set</td>
<td>251</td>
</tr>
<tr>
<td>Deleting a configuration set</td>
<td>251</td>
</tr>
<tr>
<td>Managing default configuration sets</td>
<td>252</td>
</tr>
<tr>
<td>Managing event destinations</td>
<td>254</td>
</tr>
<tr>
<td>Updating an event destination</td>
<td>255</td>
</tr>
<tr>
<td>Deleting an event destination</td>
<td>255</td>
</tr>
<tr>
<td>Enabling or disabling an event destination</td>
<td>255</td>
</tr>
<tr>
<td>Managing IP pools</td>
<td>256</td>
</tr>
<tr>
<td>Assigning an IP pool to a configuration set</td>
<td>256</td>
</tr>
<tr>
<td>Modifying IP pool assignments</td>
<td>256</td>
</tr>
<tr>
<td>Specifying configuration sets in email</td>
<td>257</td>
</tr>
<tr>
<td>Exporting reputation metrics</td>
<td>257</td>
</tr>
<tr>
<td>Enabling the exporting of reputation metrics for a configuration set</td>
<td>257</td>
</tr>
<tr>
<td>Disabling the exporting of reputation metrics for a configuration set</td>
<td>258</td>
</tr>
<tr>
<td>Monitoring your sending activity</td>
<td>259</td>
</tr>
<tr>
<td>Monitoring using the console</td>
<td>261</td>
</tr>
<tr>
<td>Monitor using the API</td>
<td>261</td>
</tr>
<tr>
<td>Calling the GetSendStatistics API operation using the AWS CLI</td>
<td>262</td>
</tr>
<tr>
<td>Calling the GetSendStatistics operation programmatically</td>
<td>262</td>
</tr>
<tr>
<td>Monitoring using notifications</td>
<td>265</td>
</tr>
<tr>
<td>Important considerations</td>
<td>265</td>
</tr>
<tr>
<td>Notifications through email</td>
<td>265</td>
</tr>
<tr>
<td>Amazon SNS notifications</td>
<td>268</td>
</tr>
<tr>
<td>Monitoring sending using event publishing</td>
<td>288</td>
</tr>
<tr>
<td>How event publishing works</td>
<td>288</td>
</tr>
<tr>
<td>How to use event publishing</td>
<td>289</td>
</tr>
<tr>
<td>Event publishing terminology</td>
<td>289</td>
</tr>
<tr>
<td>Setting up event publishing</td>
<td>290</td>
</tr>
<tr>
<td>Working with event data</td>
<td>299</td>
</tr>
<tr>
<td>Tutorials</td>
<td>341</td>
</tr>
<tr>
<td>Viewing error metrics</td>
<td>237</td>
</tr>
<tr>
<td>Managing IP address filters</td>
<td>236</td>
</tr>
<tr>
<td>Managing receipt rules</td>
<td>234</td>
</tr>
<tr>
<td>Managing receipt rule sets</td>
<td>232</td>
</tr>
<tr>
<td>Viewing the details of a receipt rule set</td>
<td>231</td>
</tr>
<tr>
<td>Deleting a receipt rule set</td>
<td>231</td>
</tr>
<tr>
<td>Creating receipt rules set</td>
<td>219</td>
</tr>
<tr>
<td>Creating receipt rule sets</td>
<td>219</td>
</tr>
<tr>
<td>Creating a receipt rule set</td>
<td>219</td>
</tr>
<tr>
<td>Creating a configuration set</td>
<td>249</td>
</tr>
<tr>
<td>Creating configuration sets</td>
<td>249</td>
</tr>
<tr>
<td>Creating a configuration set</td>
<td>249</td>
</tr>
<tr>
<td>Creating configuration sets</td>
<td>249</td>
</tr>
<tr>
<td>Viewing a list of your configuration sets</td>
<td>250</td>
</tr>
<tr>
<td>Viewing the details of a configuration set</td>
<td>251</td>
</tr>
<tr>
<td>Deleting a configuration set</td>
<td>251</td>
</tr>
<tr>
<td>Managing default configuration sets</td>
<td>252</td>
</tr>
<tr>
<td>Managing event destinations</td>
<td>254</td>
</tr>
<tr>
<td>Updating an event destination</td>
<td>255</td>
</tr>
<tr>
<td>Deleting an event destination</td>
<td>255</td>
</tr>
<tr>
<td>Enabling or disabling an event destination</td>
<td>255</td>
</tr>
<tr>
<td>Managing IP pools</td>
<td>256</td>
</tr>
<tr>
<td>Assigning an IP pool to a configuration set</td>
<td>256</td>
</tr>
<tr>
<td>Modifying IP pool assignments</td>
<td>256</td>
</tr>
<tr>
<td>Specifying configuration sets in email</td>
<td>257</td>
</tr>
<tr>
<td>Exporting reputation metrics</td>
<td>257</td>
</tr>
<tr>
<td>Enabling the exporting of reputation metrics for a configuration set</td>
<td>257</td>
</tr>
<tr>
<td>Disabling the exporting of reputation metrics for a configuration set</td>
<td>258</td>
</tr>
<tr>
<td>Monitoring your sending activity</td>
<td>259</td>
</tr>
<tr>
<td>Monitoring using the console</td>
<td>261</td>
</tr>
<tr>
<td>Monitor using the API</td>
<td>261</td>
</tr>
<tr>
<td>Calling the GetSendStatistics API operation using the AWS CLI</td>
<td>262</td>
</tr>
<tr>
<td>Calling the GetSendStatistics operation programmatically</td>
<td>262</td>
</tr>
<tr>
<td>Monitoring using notifications</td>
<td>265</td>
</tr>
<tr>
<td>Important considerations</td>
<td>265</td>
</tr>
<tr>
<td>Notifications through email</td>
<td>265</td>
</tr>
<tr>
<td>Amazon SNS notifications</td>
<td>268</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Spamtraps</td>
<td>500</td>
</tr>
<tr>
<td>Manual Investigations</td>
<td>502</td>
</tr>
<tr>
<td>DNS Blackhole List (DNSBL)</td>
<td>503</td>
</tr>
<tr>
<td>DNSBL FAQ Q1</td>
<td>504</td>
</tr>
<tr>
<td>DNSBL FAQ Q2</td>
<td>504</td>
</tr>
<tr>
<td>DNSBL FAQ Q3</td>
<td>504</td>
</tr>
<tr>
<td>DNSBL FAQ Q4</td>
<td>504</td>
</tr>
<tr>
<td>DNSBL FAQ Q5</td>
<td>505</td>
</tr>
<tr>
<td>DNSBL FAQ Q6</td>
<td>505</td>
</tr>
<tr>
<td>Email metrics</td>
<td>506</td>
</tr>
<tr>
<td>General</td>
<td>506</td>
</tr>
<tr>
<td>Open Tracking</td>
<td>507</td>
</tr>
<tr>
<td>Click Tracking</td>
<td>508</td>
</tr>
<tr>
<td>Resources</td>
<td>510</td>
</tr>
<tr>
<td>Appendix</td>
<td>511</td>
</tr>
<tr>
<td>Header fields</td>
<td>511</td>
</tr>
<tr>
<td>Unsupported attachment types</td>
<td>513</td>
</tr>
<tr>
<td>Document history</td>
<td>514</td>
</tr>
</tbody>
</table>
What is Amazon SES Classic

Amazon SES Classic refers to the original Amazon SES console, APIs, and documentation which are available for a limited time until they are deprecated. This is the original SES Developer Guide now referred to as the SES Classic Developer Guide. This change in terminology is to more clearly differentiate the original from the new Amazon SES Dev Guide that supports the new Amazon SES console.

The new Amazon SES console is now available with improved and more powerful features, such as Configuration Sets, which allow you to do everything you do in Amazon SES Classic, but with more options, increased flexibility, and efficient workflow. To glance at a quick overview of these new features, see the new Amazon SES. We encourage you to explore and take advantage of these improvements today by using the new Amazon SES console and the new Amazon SES API (referred to as V2). You can read about them in the new Amazon SES Developer Guide.

Features of Amazon SES

Amazon SES is an email platform that provides an easy, cost-effective way for you to send and receive email using your own email addresses and domains.

For example, you can send marketing emails such as special offers, transactional emails such as order confirmations, and other types of correspondence such as newsletters. When you use Amazon SES to receive mail, you can develop software solutions such as email autoresponders, email unsubscribe systems, and applications that generate customer support tickets from incoming emails.

For both Amazon SES and the new Amazon SES, you only pay for what you use. For more information, see Amazon SES Pricing.

Why use Amazon SES?

Building a large-scale email solution is often a complex and costly challenge for a business. You must deal with infrastructure challenges such as email server management, network configuration, and IP address reputation. Additionally, many third-party email solutions require contract and price negotiations, as well as significant up-front costs. Amazon SES eliminates these challenges and enables you to benefit from the years of experience and sophisticated email infrastructure Amazon.com has built to serve its own large-scale customer base.

Amazon SES and other AWS services

Amazon SES integrates seamlessly with other AWS products. For example, you can:

- Add email-sending capabilities to any application. If your application runs in Amazon Elastic Compute Cloud (Amazon EC2), you can use Amazon SES to send 62,000 emails every month at no additional charge. You can send email from Amazon EC2 by using an AWS SDK, by using the Amazon SES SMTP interface (p. 85), or by making calls directly to the Amazon SES API.
- Use AWS Elastic Beanstalk to create an email-enabled application such as a program that uses Amazon SES to send a newsletter to customers.
- Set up Amazon Simple Notification Service (Amazon SNS) to notify you of your emails that bounced, produced a complaint, or were successfully delivered to the recipient's mail server. When you use Amazon SES to receive emails, your email content can be published to Amazon SNS topics.
In this guide

This guide contains the following sections:

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sending email (p. 5)</td>
<td>Describes how you can send email using Amazon SES.</td>
</tr>
<tr>
<td>Receiving email (p. 203)</td>
<td>Describes how you can receive email using Amazon SES.</td>
</tr>
<tr>
<td>Controlling access (p. 397)</td>
<td>Shows you how to use Amazon SES with AWS Identity and Access Management (IAM) to specify which Amazon SES API actions a user can perform on which Amazon SES resources.</td>
</tr>
<tr>
<td>Logging API calls (p. 403)</td>
<td>Provides a list of Amazon SES APIs that can be logged using AWS CloudTrail.</td>
</tr>
<tr>
<td>Types of credentials (p. 16)</td>
<td>Explains the types of credentials that you might use with Amazon SES, and when you might use them.</td>
</tr>
<tr>
<td>Using the API (p. 407)</td>
<td>Describes how to interact with the Amazon SES API.</td>
</tr>
<tr>
<td>Regions (p. 457)</td>
<td>Contains information about using Amazon SES in multiple AWS Regions.</td>
</tr>
<tr>
<td>Quotas (p. 461)</td>
<td>Provides a list of quotas that apply to Amazon SES.</td>
</tr>
<tr>
<td>Resources (p. 510)</td>
<td>Lists resources that you may find useful as you work with Amazon SES.</td>
</tr>
<tr>
<td>Appendix (p. 511)</td>
<td>Provides supplementary information about header fields, unsupported attachment types, and scripts.</td>
</tr>
</tbody>
</table>
Amazon SES Quick start

This procedure leads you through the steps to sign up for AWS, verify your email address, send your first email, consider how you will handle bounces and complaints, and move out of the Amazon Simple Email Service (Amazon SES) sandbox.

Use this procedure if you:

- Are just experimenting with Amazon SES.
- Want to send some test emails without doing any programming.
- Want to get set up in as few steps as possible.

Step 1: Sign up for AWS

Before you can use Amazon SES, you need to sign up for AWS. When you sign up for AWS, your account is automatically signed up for all AWS services.

For instructions, see Signing up for AWS (p. 46).

Step 2: Verify your email address

Before you can send email from your email address through Amazon SES, you need to show Amazon SES that you own the email address by verifying it.

For instructions, see Verifying email addresses in Amazon SES (p. 47).

Step 3: Send your first email

You can send an email simply by using the Amazon SES console. As a new user, your account is in a test environment called the sandbox, so you can only send email to and from email addresses that you have verified.

For instructions, see Send an email using the Amazon SES console (p. 19).

Step 4: Consider how you will handle bounces and complaints

Before the next step, you need to think about how you will handle bounces and complaints. If you are sending to a small number of recipients, your process can be as simple as examining the bounce and complaint feedback that you receive by email, and then removing those recipients from your mailing list.

Step 5: Move out of the Amazon SES sandbox

To be able to send emails to unverified email addresses and to raise the number of emails you can send per day and how fast you can send them, your account needs to be moved out of the sandbox. This process involves opening an SES Sending Limits Increase case in Support Center.
For more information about the sandbox restrictions and how to apply to move out of the sandbox, see Moving out of the Amazon SES sandbox (p. 71).

Next steps

- After you send a few test emails to yourself, use the Amazon SES mailbox simulator for further testing because emails to the mailbox simulator do not count towards your sending quota or your bounce and complaint rates. For more information on the mailbox simulator, see Testing email sending in Amazon SES (p. 181).
- Monitor your sending activity, such as the number of emails that you have sent and the number that have bounced or received complaints. For more information, see Monitoring your Amazon SES sending activity (p. 259).
- Verify entire domains so that you can send email from any email address in your domain without verifying addresses individually. For more information, see Verifying domains in Amazon SES (p. 59).
- Increase the chance that your emails will be delivered to your recipients' inboxes instead of junk boxes by authenticating your emails. For more information, see Authenticating your email in Amazon SES (p. 129).
Sending email with Amazon SES

When you send an email, you are sending it through some type of outbound email server. That email server might be provided by your Internet service provider (ISP), your company’s IT department, or you might have set it up yourself. The email server accepts your email content, formats it to comply with email standards, and then sends the email out over the Internet. The email may pass through other servers until it eventually reaches a receiver (an entity, such as an ISP, that receives the email on behalf of the recipient). The receiver then delivers the email to the recipient. The following diagram illustrates the basic email-sending process.

When you use Amazon SES, Amazon SES becomes your outbound email server. You can also keep your existing email server and configure it to send your outgoing emails through Amazon SES so that you don’t have to change any settings in your email clients. The following diagram shows where Amazon SES fits in to the email sending process.

A sender can generate the email content in different ways. A sender can create the email by using an email client application, or use a program that automatically generates emails, like an application that sends order confirmations in response to purchase transactions.

How do I send emails using Amazon SES?

There are several ways that you can send an email by using Amazon SES. You can use the Amazon SES console, the Simple Mail Transfer Protocol (SMTP) interface, or you can call the Amazon SES API.

- **Amazon SES console**—This method is the quickest way to set up your system and send a couple of test emails, but once you are ready to start your email campaign, you will use the console primarily to monitor your sending activity. For example, you can quickly view the number of emails that you have sent and the number of bounces and complaints that you have received.

- **SMTP Interface**—There are two ways to access Amazon SES through the SMTP interface. The first way, which requires no coding, is to configure any SMTP-enabled software to send email through Amazon SES. For example, you can configure your existing email client or software program to connect to the Amazon SES SMTP endpoint instead of your current outbound email server.

  The second way is to use an SMTP-compatible programming language such as Java and access the Amazon SES SMTP interface by using the language’s built-in SMTP functions and data types.

- **Amazon SES API**—You can call the Amazon SES Query API directly through HTTPS, or you can use the AWS Command Line Interface, the AWS Tools for Windows PowerShell, or an AWS SDK. The AWS SDKs wrap the low-level functionality of the Amazon SES API with higher-level data types and function calls that take care of the details for you. The AWS SDKs provide not only Amazon SES operations, but also basic AWS functionality such as request authentication, request retries, and error handling.
How do I start?

If you're new to Amazon SES, start by reading the following sections:

- **Amazon SES Quick start (p. 3)**—Shows you how to get set up and send a test email as quickly as possible.
- **Getting started sending email with Amazon SES (p. 18)**—Shows you how to send an email by using the Amazon SES console, the SMTP interface, and an AWS SDK. Examples are provided in C#, Java, and PHP.
- **Amazon SES and deliverability (p. 7)**—Explains email deliverability concepts that you should be familiar with when you use Amazon SES.
- **Amazon SES email-sending process (p. 11)**—Shows you what happens when you send an email through Amazon SES.
- **Email format and Amazon SES (p. 14)**—Reviews the format of emails and identifies the information that you need to provide to Amazon SES.

Then you can learn about sending email with Amazon SES in more detail by reading the sections listed in the following table:

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting up email (p. 46)</td>
<td>Shows you how to sign up for AWS, get your AWS access keys, download an AWS SDK, verify email addresses or domains, and move out of the Amazon SES sandbox.</td>
</tr>
<tr>
<td>Using the SMTP interface (p. 85)</td>
<td>Shows you how to get your Amazon SES SMTP credentials, connect to the Amazon SES SMTP endpoint, and provides examples of how to configure email clients and software packages to send email through Amazon SES. Also explains how to configure your existing email server to send all outgoing emails through Amazon SES.</td>
</tr>
<tr>
<td>Using the API (p. 109)</td>
<td>Shows you how to send formatted and raw emails by using the Amazon SES API. Explains how to use non-standard characters and send attachments by using the Multipurpose Internet Mail Extensions (MIME) standard when you send raw emails.</td>
</tr>
<tr>
<td>Email authentication methods (p. 129)</td>
<td>Shows you how to use DKIM with Amazon SES to show ISPs that you own the domain you are sending from.</td>
</tr>
<tr>
<td>Managing sending quotas (p. 144)</td>
<td>Describes the Amazon SES sending quotas, provides procedures for increasing them, and documents the errors you receive when you exceed them.</td>
</tr>
<tr>
<td>Using sending authorization (p. 149)</td>
<td>Shows you how to authorize other users to send emails from your identities on your behalf.</td>
</tr>
<tr>
<td>Using dedicated IP addresses (p. 173)</td>
<td><em>you decide whether to use shared IP addresses or lease dedicated IP addresses for your Amazon SES sending. Provides procedures for requesting and relinquishing dedicated IPs, and for creating pools of dedicated IPs.</em></td>
</tr>
<tr>
<td>Testing email sending (p. 181)</td>
<td>Explains how to use the Amazon SES mailbox simulator to simulate common email scenarios without affecting...</td>
</tr>
</tbody>
</table>
Amazon SES email-sending concepts

The following sections contain information about how Amazon SES sends your mail.

Topics in this section:
- Amazon SES and deliverability (p. 7)
- Amazon SES email-sending process (p. 11)
- Email format and Amazon SES (p. 14)
- Types of Amazon SES credentials (p. 16)

Amazon SES and deliverability

You want your recipients to read your emails, find them valuable, and not label them as spam. In other words, you want to maximize email deliverability—the percentage of your emails that arrive in your recipients' inboxes. This topic reviews email deliverability concepts that you should be familiar with when you use Amazon SES.

To maximize email deliverability, you need to understand email delivery issues, proactively take steps to prevent them, stay informed of the status of the emails that you send, and then improve your email-sending program, if necessary, to further increase the likelihood of successful deliveries. The following sections review the concepts behind these steps and how Amazon SES helps you through the process.
Understand email delivery issues

In most cases, your messages are delivered successfully to recipients who expect them. In some cases, however, a delivery might fail, or a recipient might not want to receive the mail that you are sending. Bounces, complaints, and the suppression list are related to these delivery issues and are described in the following sections.

Bounce

If your recipient's receiver (for example, an email provider) fails to deliver your message to the recipient, the receiver bounces the message back to Amazon SES. Amazon SES then notifies you of the bounced email through email or through Amazon Simple Notification Service (Amazon SNS), depending on how you have your system set up. For more information, see Monitoring Amazon SES email sending using notifications (p. 265).

There are hard bounces and soft bounces, as follows:

- **Hard bounce** – A persistent email delivery failure. For example, the mailbox does not exist. Amazon SES does not retry hard bounces, with the exception of DNS lookup failures. We strongly recommend that you do not make repeated delivery attempts to email addresses that hard bounce.

- **Soft bounce** – A temporary email delivery failure. For example, the mailbox is full, there are too many connections (also called throttling), or the connection times out. Amazon SES retries soft bounces multiple times. If the email still cannot be delivered, then Amazon SES stops retrying it.
Amazon SES notifies you of hard bounces and soft bounces that will no longer be retried. However, only hard bounces count toward your bounce rate and the bounce metric that you retrieve using the Amazon SES console or the GetSendStatistics API.

Bounces can also be synchronous or asynchronous. A synchronous bounce occurs while the email servers of the sender and receiver are actively communicating. An asynchronous bounce occurs when a receiver initially accepts an email message for delivery and then subsequently fails to deliver it to the recipient.

**Complaint**

Most email client programs provide a button labeled "Mark as Spam," or similar, which moves the message to a spam folder, and forwards it to the email provider. Additionally, most email providers maintain an abuse address (e.g., abuse@example.net), where users can forward unwanted email messages and request that the email provider take action to prevent them. In both of these cases, the recipient is making a complaint. If the email provider concludes that you are a spammer, and Amazon SES has a feedback loop set up with the email provider, then the email provider will send the complaint back to Amazon SES. When Amazon SES receives such a complaint, it forwards the complaint to you either by email or by using an Amazon SNS notification, depending on how you have your system set up. For more information, see Monitoring Amazon SES email sending using notifications (p. 265). We recommend that you do not make repeated delivery attempts to email addresses that generate complaints.

**Global suppression list**

The Amazon SES global suppression list is a list of recipient email addresses that have recently caused a hard bounce for any Amazon SES customer. If you try to send an email through Amazon SES to an address that is on the suppression list, the call to Amazon SES succeeds, but Amazon SES treats the email as a hard bounce instead of attempting to send it. Like any hard bounce, suppression list bounces count towards your sending quota and your bounce rate. An email address can remain on the suppression list for up to 14 days. If you are sure that the email address that you're trying to send to is valid, you can submit a suppression list removal request. For more information, see Using the Amazon SES global suppression list (p. 192).

**Be proactive**

One of the biggest issues with email on the Internet is unsolicited bulk email (spam). Email providers take extensive measures to prevent their customers from receiving spam. Amazon SES also takes steps to decrease the likelihood that email providers consider your email to be spam. Amazon SES uses verification, authentication, sending quotas, and content filtering. Amazon SES also maintains a trusted reputation with email providers and requires you to send high-quality email. Amazon SES does some of those things for you automatically (for example, content filtering); in other cases, it provides the tools (such as authentication), or guides you in the right direction (sending quotas). The following sections provide more information about each concept.

**Verification**

Unfortunately, it's possible for a spammer to falsify an email header and spoof the originating email address so that it appears as though the email originated from a different source. To maintain trust between email providers and Amazon SES, Amazon SES needs to ensure that its senders are who they say they are. You are therefore required to verify all email addresses from which you send emails through Amazon SES to protect your sending identity. You can verify email addresses by using the Amazon SES console or by using the Amazon SES API. You can also verify entire domains. For more information, see Verifying email addresses in Amazon SES (p. 47) and Verifying domains in Amazon SES (p. 59).

If your account is still in the Amazon SES sandbox, you also need to verify all recipient addresses except for addresses provided by the Amazon SES mailbox simulator. For information about getting out of the sandbox, see Moving out of the Amazon SES sandbox (p. 71). For more information about the mailbox simulator, see Testing email sending in Amazon SES (p. 181).
Authentication

Authentication is another way that you can indicate to email providers that you are who you say you are. When you authenticate an email, you provide evidence that you are the owner of the account and that your emails have not been modified in transit. In some cases, email providers refuse to forward email that is not authenticated. Amazon SES supports two methods of authentication: Sender Policy Framework (SPF) and DomainKeys Identified Mail (DKIM). For more information, see Authenticating your email in Amazon SES (p. 129).

Sending quotas

If an email provider detects sudden, unexpected spikes in the volume or rate of your emails, the email provider might suspect you are a spammer and block your emails. Therefore, every Amazon SES account has a set of sending quotas. These quotas restrict the number of emails that you can send in a 24-hour period, and the number that you can send per second. These sending quotas help protect your trustworthiness with email providers.

In most cases, if you're a brand-new user, Amazon SES lets you send a small amount of email each day. If the mail that you send is acceptable to email providers, we automatically increase this quota. Your sending quotas steadily increase over time so that you can send larger quantities of email at faster rates. You can also create an SES Sending Limits Increase case to request additional quota increases.

For more information about sending quotas and how to increase them, see Managing your Amazon SES sending quotas (p. 144).

Content filtering

Many email providers use content filtering to determine if incoming emails are spam. Content filters look for questionable content and block the email if the email fits the profile of spam. Amazon SES uses content filters also. When your application sends a request to Amazon SES, Amazon SES assembles an email message on your behalf and then scans the message header and body to determine if they contain content that email providers might consider spam. If your messages look like spam to the content filters that Amazon SES uses, your reputation with Amazon SES will be negatively affected.

Amazon SES also scans all messages for viruses. If a message contains a virus, Amazon SES doesn't attempt to deliver the message to the recipient's mail server.

Reputation

When it comes to email sending, reputation—a measure of confidence that an IP address, email address, or sending domain is not the source of spam—is important. Amazon SES maintains a strong reputation with email providers so that they deliver your email to your recipients' inboxes. Similarly, you need to maintain a trusted reputation with Amazon SES. You build your reputation with Amazon SES by sending high-quality content. When you send high-quality content, your reputation becomes more trusted over time and Amazon SES increases your sending quotas. Excessive bounces and complaints negatively impact your reputation and can cause Amazon SES to reduce the sending quotas for your account, or terminate your Amazon SES account.

One way to help maintain your reputation is to use the mailbox simulator when you test your system, instead of sending to email addresses that you have created yourself. Emails to the mailbox simulator do not count toward your bounce and complaint metrics. For more information about the mailbox simulator, see Testing email sending in Amazon SES (p. 181).

High-quality email

High-quality email is email that recipients find valuable and want to receive. Value means different things to different recipients and can come in the form of offers, order confirmations, receipts,
Stay informed

Whether your deliveries fail, your recipients complain about your emails, or Amazon SES successfully delivers an email to a recipient's mail server, Amazon SES helps you to track down the issue by providing notifications and by enabling you to easily monitor your usage statistics.

Notifications

When an email bounces, the email provider notifies Amazon SES, and Amazon SES notifies you. Amazon SES notifies you of hard bounces and soft bounces that Amazon SES will no longer retry. Many email providers also forward complaints, and Amazon SES sets up complaint feedback loops with the major email providers so you don't have to. Amazon SES can notify you of bounces, complaints, and successful deliveries in two ways: you can set your account up to receive notifications through Amazon SNS, or you can receive notifications by email (bounces and complaints only). For more information, see Monitoring Amazon SES email sending using notifications (p. 265).

Usage statistics

Amazon SES provides usage statistics so that you can view your failed deliveries to determine and resolve the root causes. You can view your usage statistics by using the Amazon SES console or by calling the Amazon SES API. You can view how many deliveries, bounces, complaints, and virus-infected rejected emails you have, and you can also view your sending quotas to ensure that you stay within them.

Improve your email-sending program

If you are getting large numbers of bounces and complaints, it's time to reassess your email-sending strategy. Remember that excessive bounces, complaints, and attempts to send low-quality email constitute abuse and put your AWS account at risk of termination. Ultimately, you need to be sure that you use Amazon SES to send high-quality emails and to only send emails to recipients who want to receive them.

Amazon SES email-sending process

This topic describes what happens when you send an email with Amazon SES, and the various outcomes that can occur after the email is sent. The following figure is a high-level overview of the sending process:

1. A client application, acting as an email sender, makes a request to Amazon SES to send email to one or more recipients.
2. If the request is valid, Amazon SES accepts the email.
3. Amazon SES sends the message over the Internet to the recipient's receiver. Once the message
   is passed to Amazon SES, it is usually sent immediately, with the first delivery attempt normally
   occurring within milliseconds.
4. At this point, there are different possibilities. For example:
   a. The ISP successfully delivers the message to the recipient's inbox.
   b. The recipient's email address does not exist, so the ISP sends a bounce notification to Amazon
      SES. Amazon SES then forwards the notification to the sender.
   c. The recipient receives the message but considers it to be spam and registers a complaint with
      the ISP. The ISP, which has a feedback loop set up with Amazon SES, sends the complaint to
      Amazon SES, which then forwards it to the sender.

The following sections review the individual possible outcomes after a sender sends an email request to
Amazon SES and after Amazon SES sends an email message to the recipient.

**After a sender sends an email request to Amazon SES**

When the sender makes a request to Amazon SES to send an email, the call may succeed or fail. The
following sections describe what happens in each case.

**Successful sending request**

If the request to Amazon SES succeeds, Amazon SES returns a success response to the sender. This
message includes the message ID, a string of characters that uniquely identifies the request. You can use
the message ID to identify the sent email or to track problems encountered during sending. Amazon SES
then assembles an email message based on the request parameters, scans the message for questionable
content and viruses and then sends it out over the Internet using Simple Mail Transfer Protocol (SMTP).
Your message is usually sent immediately; the first delivery attempt typically occurs within milliseconds.

**Note**

If Amazon SES accepts the sender's request and then determines that the message contains
a virus, Amazon SES stops processing the message and doesn't attempt to deliver it to the
recipient's mail server.

**Failed sending request**

If the sender's email-sending request to Amazon SES fails, Amazon SES responds to the sender with an
error and drops the email. The request could fail for several reasons. For example, the request may not
be formatted properly or the email address may not have been verified by the sender.

The method through which you can determine if the request has failed depends on how you call Amazon
SES. The following are examples of how errors and exceptions are returned:

- If you are calling Amazon SES through the Query (HTTPS) API (SendEmail or SendRawEmail), the
  actions will return an error. For more information, see the Amazon Simple Email Service API Reference.
- If you are using an AWS SDK for a programming language that uses exceptions, the call to Amazon SES
  will throw a MessageRejectedException. (The name of the exception may vary slightly depending on the
  SDK.)
- If you are using the SMTP interface, then the sender receives an SMTP response code, but how the
  error is conveyed depends on the sender's client. Some clients may display an error code; others may
  not.

For information about errors that can occur when you send an email with Amazon SES, see Amazon SES
email sending errors (p. 476).
After Amazon SES sends an email

If the sender's request to Amazon SES succeeds, then Amazon SES sends the email and one of the following outcomes occurs:

- **Successful delivery and the recipient does not object to the email**—The email is accepted by the ISP, and the ISP delivers the email to the recipient. A successful delivery is shown in the following figure.

![Successful delivery](image)

- **Hard bounce**—The email is rejected by the ISP because of a persistent condition or rejected by Amazon SES because the email address is on the Amazon SES suppression list. An email address is on the Amazon SES suppression list if it has recently caused a hard bounce for any Amazon SES customer. A hard bounce with an ISP can occur because the recipient's address is invalid. A hard bounce notification is sent from the ISP back to Amazon SES, which notifies the sender through email or through Amazon Simple Notification Service (Amazon SNS), depending on the sender's setup. Amazon SES notifies the sender of suppression list bounces by the same means. The path of a hard bounce from an ISP is shown in the following figure.

![Hard bounce](image)

- **Soft bounce**—The ISP cannot deliver the email to the recipient because of a temporary condition, such as the ISP is too busy to handle the request or the recipient's mailbox is full. A soft bounce can also occur if the domain does not exist. The ISP sends a soft bounce notification back to Amazon SES, or, in the case of a nonexistent domain, Amazon SES cannot find an email server for the domain. In either case, Amazon SES retries the email for an extended period of time. If Amazon SES cannot deliver the email in that time period, it sends you a bounce notification through email or through Amazon SNS. If Amazon SES can deliver the email to the recipient during a retry, the delivery is successful. A soft bounce is shown in the following figure. In this case, Amazon SES retries sending the email, and the ISP is eventually able to deliver it to the recipient.

![Soft bounce](image)

- **Complaint**—The email is accepted by the ISP and delivered to the recipient, but the recipient considers the email to be spam and clicks a button such as "Mark as spam" in his or her email client. If Amazon SES has a feedback loop set up with the ISP, then a complaint notification is sent to Amazon SES, which forwards the complaint notification to the sender. Most ISPs do not provide the email address of the recipient who submitted the complaint, so the complaint notification from Amazon SES provides the sender a list of recipients who might have sent the complaint, based on the recipients of the original message and the ISP from which Amazon SES received the complaint. The path of a complaint is shown in the following figure.

![Complaint](image)

- **Auto response**—The email is accepted by the ISP, and the ISP delivers it to the recipient. The ISP then sends an automatic response such as an out-of-the-office (OOTO) message to Amazon SES. Amazon SES forwards the auto response notification to the sender. An auto response is shown in the following figure.
Make sure that your Amazon SES-enabled program does not retry sending messages that generate an auto response.

**Tip**
You can use the Amazon SES mailbox simulator to test a successful delivery, bounce, complaint, OOTO, or what happens when an address is on the suppression list. For more information, see Testing email sending in Amazon SES (p. 181).

## Email format and Amazon SES

When a client makes a request to Amazon SES, Amazon SES constructs an email message compliant with the Internet Message Format specification (RFC 5322). An email consists of a header, a body, and an envelope, as described below.

- **Header**—Contains routing instructions and information about the message. Examples are the sender's address, the recipient's address, the subject, and the date. The header is analogous to the information at the top of a postal letter, though it can contain many other types of information, such as the format of the message.

- **Body**—Contains the text of the message itself.

- **Envelope**—Contains the actual routing information that is communicated between the email client and the mail server during the SMTP session. This email envelope information is analogous to the information on a postal envelope. The routing information of the email envelope is usually the same as the routing information in the email header, but not always. For example, when you send a blind carbon copy (BCC), the actual recipient address (derived from the envelope) is not the same as the "To" address that is displayed in the recipient's email client, which is derived from the header.

The following is a simple example of an email. The header is followed by a blank line and then the body of the email. The envelope isn't shown because it is communicated between the client and the mail server during the SMTP session, rather than a part of the email itself.

```
Received: from abc.smtp-out.amazonses.com (123.45.67.89) by in.example.com (87.65.43.210);
Fri, 17 Dec 2010 14:26:22
From: "Andrew" <andrew@example.com>
To: "Bob" <bob@example.com>
Date: Fri, 17 Dec 2010 14:26:21 -0800
Subject: Hello
Message-ID: <61967230-7A45-4A9D-BEC9-87CBCF2211C9@example.com>
Accept-Language: en-US
Content-Language: en-US
Content-Type: text/plain; charset="us-ascii"
Content-Transfer-Encoding: quoted-printable
MIME-Version: 1.0

Hello, I hope you are having a good day.

-Andrew
```

The following sections review email headers and bodies and identify the information that you need to provide when you use Amazon SES.
Email header

There is one header per email message. Each line of the header contains a field followed by a colon
followed by a field body. When you read an email in an email client, the email client typically displays the
values of the following header fields:

- **To**—The email addresses of the message's recipients.
- **CC**—The email addresses of the message's carbon copy recipients.
- **From**—The email address from which the email is sent.
- **Subject**—A summary of the message topic.
- **Date**—The time and date the email is sent.

There are many additional header fields that provide routing information and describe the content of
the message. Email clients typically do not display these fields to the user. For a full list of the header
fields that Amazon SES accepts, see Header fields (p. 511). When you use Amazon SES, you particularly
need to understand the difference between "From," "Reply-To," and "Return-Path" header fields. As
noted previously, the "From" address is the email address of the message sender, whereas "Reply-To" and
"Return-Path" are as follows:

- **Reply-To**—The email address to which replies will be sent. By default, replies are sent to the original
  sender's email address.
- **Return-Path**—The email address to which message bounces and complaints should be sent. "Return-
  Path" is sometimes called "envelope from," "envelope sender," or "MAIL FROM."

  **Note**
  When you use Amazon SES, we recommend that you always set the "Return-Path" parameter
  so that you can be aware of bounces and take corrective action if they occur.

To easily match a bounced message with its intended recipient, you can use Variable Envelope Return
Path (VERP). With VERP, you set a different "Return-Path" for each recipient, so that if the message
bounces back, you automatically know which recipient it bounced from, rather than having to open the
bounce message and parse it.

Email body

The email body contains the text of the message. The body can be sent in the following formats:

- **HTML**—If the recipient's email client can interpret HTML, the body can include formatted text and
  hyperlinks
- **Plain text**—If the recipient's email client is text-based, the body must not contain any nonprintable
  characters.
- **Both HTML and plain text**—When you use both formats to send the same content in a single
  message, the recipient's email client decides which to display, based upon its capabilities.

If you are sending an email message to a large number of recipients, then it makes sense to send it
in both HTML and text. Some recipients will have HTML-enabled email clients, so that they can click
embedded hyperlinks in the message. Recipients using text-based email clients will need you to include
URLs that they can copy and open using a web browser.

Email information you need to provide to Amazon SES

When you send an email with Amazon SES, the email information you need to provide depends on how
you call Amazon SES. You can provide a minimal amount of information and have Amazon SES take care
of all of the formatting for you. Or, if you want to do something more advanced like send an attachment,
you can provide the raw message yourself. The following sections review what you need to provide when
you send an email by using the Amazon SES API, the Amazon SES SMTP interface, or the Amazon SES
console.

Amazon SES API

If you call the Amazon SES API directly, you call either the SendEmail or the SendRawEmail API. The
amount of information you need to provide depends on which API you call.

• The SendEmail API requires you to provide only a source address, destination address, message
subject, and a message body. You can optionally provide "Reply-To" addresses. When you call this
API, Amazon SES automatically assembles a properly formatted multi-part Multipurpose Internet
Mail Extensions (MIME) email message optimized for display by email client software. For more
information, see Sending formatted email using the Amazon SES API (p. 109).

• The SendRawEmail API provides you the flexibility to format and send your own raw email message
by specifying headers, MIME parts, and content types. SendRawEmail is typically used by advanced
users. You need to provide the body of the message and all header fields that are specified as required
in the Internet Message Format specification (RFC 5322). For more information, see Sending raw email
using the Amazon SES API (p. 110).

If you use an AWS SDK to call the Amazon SES API, you provide the information listed above to the
corresponding functions (for example, SendEmail and SendRawEmail for Java).

For more information about sending email using the Amazon SES API, see Using the Amazon SES API to
send email (p. 109).

Amazon SES SMTP interface

When you access Amazon SES through the SMTP interface, your SMTP client application assembles
the message, so the information you need to provide depends on the application you are using. At a
minimum, the SMTP exchange between a client and a server requires a source address, a destination
address, and message data. If you are using the SMTP interface and have feedback forwarding enabled,
then your bounces, complaints, and delivery notifications are sent to the “MAIL FROM” address. Any
"Reply-To" address that you specify is not used.

For more information about sending email using the Amazon SES SMTP interface, see Using the Amazon
SES SMTP interface to send email (p. 85).

Amazon SES console

When you send an email by using the Amazon SES console, the amount of information you need to
provide depends on whether you choose to send a formatted or raw email.

• To send a formatted email, you need to provide a source address, a destination address, a message
subject, and a message body. Amazon SES automatically assembles a properly formatted multi-part
MIME email message optimized for display by email client software. You can also specify a reply-to
and a return path field.

• To send a raw email, you provide the source address, a destination address, and the message content,
which must contain the body of the message and all header fields that are specified as required in the
Internet Message Format specification (RFC 5322).

Types of Amazon SES credentials

To interact with Amazon Simple Email Service (Amazon SES), you use security credentials to verify who
you are and whether you have permission to interact with Amazon SES. There are different types of
credentials, and the credentials you use depend on what you want to do. For example, you use AWS
access keys when you send an email using the Amazon SES API, and SMTP credentials when you send an email using the Amazon SES SMTP interface.

The following table lists the types of credentials you might use with Amazon SES, depending on what you are doing.

<table>
<thead>
<tr>
<th>If you want to access the...</th>
<th>Use these credentials</th>
<th>What the credentials consist of</th>
<th>How to get the credentials</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon SES API</td>
<td>AWS access keys</td>
<td>Access key ID and secret access key</td>
<td>See Access Keys in the AWS General Reference.</td>
</tr>
<tr>
<td>(You might access the Amazon SES API directly, or indirectly through an AWS SDK, the AWS Command Line Interface, or the AWS Tools for Windows PowerShell.)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Amazon SES SMTP interface</td>
<td>SMTP credentials</td>
<td>User name and password</td>
<td>See Obtaining your Amazon SES SMTP credentials (p. 86).</td>
</tr>
</tbody>
</table>

**Note**
For security best practice, use AWS Identity and Access Management (IAM) user access keys instead of AWS account access keys. Your AWS account credentials grant full access to all your AWS resources, so you should store them in a safe place and instead use IAM user credentials for day-to-day interaction with AWS. For more information, see Root Account Credentials vs. IAM User Credentials in the AWS General Reference.

Although your Amazon SES SMTP credentials are different than your AWS access keys and IAM user access keys, Amazon SES SMTP credentials are actually a type of IAM credentials. An IAM user can create Amazon SES SMTP credentials, but the root account owner must ensure that the IAM user’s policy gives them permission to access the following IAM actions: “iam:ListUsers”, “iam:CreateUser”, “iam:PassChangePassword”, “iam:PassResetPassword”. See Root Account Credentials vs. IAM User Credentials in the AWS General Reference.
Getting started

If you want to access the... | Use these credentials | What the credentials consist of | How to get the credentials |
--- | --- | --- | --- |
Amazon SES console | IAM user name and password OR Email address and password | IAM user name and password OR Email address and password | "iam:CreateAccessKey", and "iam:PutUserPolicy". See IAM User Name and Password and Email Address and Password of the AWS General Reference. Note For security best practice, use an IAM user name and password instead of an email address and password. The email address and password combination are for your AWS account, so you should store them in a safe place instead of using them for day-to-day interaction with AWS. For more information, see Root Account Credentials vs. IAM User Credentials in the AWS General Reference. |

For more information about different types of AWS security credentials (except for SMTP credentials, which are used only for Amazon SES), see AWS Security Credentials in the AWS General Reference.

Getting started sending email with Amazon SES

This getting started tutorial provides step-by-step instructions for you to set up Amazon Simple Email Service (Amazon SES) and send an email. First, review the information in Before you begin with Amazon SES (p. 19). Then, send an email in one of the following ways.

Using the Amazon SES console

Use this method if you want to get started sending test emails through Amazon SES with minimal setup. When you are ready to start your production email sending campaign, use one of the other sending methods and use the Amazon SES console primarily to monitor your sending activity.

To start this tutorial, go to Send an email using the Amazon SES console (p. 19).

Using Simple Mail Transfer Protocol (SMTP)

Use this method if you want to send email through the Amazon SES SMTP interface with or without programming as follows:
• Enable an application to send email through Amazon SES by using a programming language that supports SMTP. Examples are provided in C#, Java, and PHP. To start this tutorial, go to Send an email by accessing the Amazon SES SMTP interface programmatically (p. 21).

• Set up your mail server to forward mail to Amazon SES, or configure your email client or SMTP-enabled software package to send email through Amazon SES. Examples are provided for Postfix, Sendmail, and Exim mail servers. To start this tutorial, go to Configuring Your Existing Email Server or SMTP-Enabled Application to Send Email Through Amazon SES (p. 30).

For introductory information on both SMTP sending methods, see Send an email through Amazon SES using SMTP (p. 20).

Using an AWS SDK

Use this method to call the Amazon SES API using libraries that handle the details of the underlying Amazon SES Query interface. Examples are provided in C#, Java, PHP, Ruby, and Python. To start this tutorial, go to Send an email through Amazon SES using an AWS SDK (p. 30).

Before you begin with Amazon SES

Before you start, you need to set up Amazon SES. Whether you send an email by using the Amazon SES console, the SMTP interface, or the Amazon SES API, you need to:

• Sign up for AWS—Before you can use Amazon SES or other AWS services, you need to create an AWS account. For information, see Signing up for AWS (p. 46).

• Verify your email address or domain—To send emails using Amazon SES, you always need to verify your "From" address to show that you own it. If you are in the sandbox, you also need to verify your "To" addresses. You can verify email addresses or entire domains. For information, see Verifying identities in Amazon SES (p. 47).

• Provide your account details—Keep your account up-to-date by editing your account details. Providing your account details also allows you to request to move out of the sandbox. When in the sandbox, you can only send email to the Amazon SES mailbox simulator and verified email addresses or domains. For information, see Moving out of the Amazon SES sandbox (p. 71).

This list contains the setup tasks that are mandatory for all email sending methods. Additional setup tasks that are specific to the email sending method are provided in the corresponding getting started section. To see a complete list of all setup tasks, see Setting up email with Amazon SES (p. 46).

Send an email using the Amazon SES console

The easiest way to send an email with Amazon SES is to use the Amazon SES console. Because the console requires you to manually enter information, you typically only use it to send test emails. After you get started with Amazon SES, you will most likely send your emails using either the Amazon SES SMTP interface or API, but the console is useful for monitoring your sending activity.

Important
In this getting started tutorial, you send an email to yourself so that you can check to see if you received it. For further experimentation or load testing, use the Amazon SES mailbox simulator. Emails that you send to the mailbox simulator do not count toward your sending quota or your bounce and complaint rates. For more information, see Testing email sending in Amazon SES (p. 181).

Before you follow these steps, make sure you review the setup instructions in Before you begin with Amazon SES (p. 19).
To send an email message from the Amazon SES console

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.

   **Note**
   If you are not currently signed in to your AWS account, this link takes you to a sign-in page. After you sign in, you are directed to the Amazon SES console.

2. In the navigation pane on the left side of the Amazon SES console, under Identity Management, choose Email Addresses to view the email address that you verified in Verifying email addresses in Amazon SES (p. 47).

3. In the list of identities, check the box next to email address that you have verified.

4. Choose Send a Test Email.

5. For Send Test Email, choose the Email Format. The two choices are as follows:

   - **Formatted**—This is the simplest option. Choose this option if you simply want to type the text of your message into the Body text box. When you send the email, Amazon SES puts the text into email format for you.

   - **Raw**—Choose this option if you want to send a more complex message, such as a message that includes HTML or an attachment. Because of this flexibility, you need to format the message, as described in Sending raw email using the Amazon SES API (p. 110), yourself, and then paste the entire formatted message, including the headers, into the Body text box. You can use the following example, which contains HTML, to send a test email using the Raw email format. Copy and paste this message in its entirety into the Body text box. Ensure that there is not a blank line between the MIME-Version header and the Content-Type header; a blank line between these two lines causes the email to be formatted as plain text instead of HTML.

   ```
   Subject: Amazon SES Raw Email Test
   MIME-Version: 1.0
   Content-Type: text/html
   <!DOCTYPE html>
   <html>
   <body>
   <h1>This text should be large, because it is formatted as a header in HTML.</h1>
   <p>Here is a formatted link: <a href="https://docs.aws.amazon.com/ses/latest/DeveloperGuide/Welcome.html">Amazon Simple Email Service Developer Guide</a>.</p>
   </body>
   </html>
   ```

   6. For Send Test Email, fill out the rest of the fields. If you are still in the Amazon SES sandbox, make sure that the address in the To field is a verified email address. For more information, see Verifying email addresses in Amazon SES (p. 47).

   7. Choose Send Test Email.

   8. Sign in to the email client of the address you sent the email to. You will find the message that you sent.

Send an email through Amazon SES using SMTP

To send an email using the Amazon SES SMTP interface, you can use an SMTP-enabled programming language, email server, or application. Before you start, review the instructions in Before you begin with Amazon SES (p. 19). You also need to get the following additional information:
• Your Amazon SES SMTP username and password, which enable you to connect to the Amazon SES SMTP endpoint. To get your Amazon SES SMTP username and password, see Obtaining your Amazon SES SMTP credentials (p. 86).

  **Important**
  Your SMTP credentials are different from your AWS credentials. For more information about credentials, see Types of Amazon SES credentials (p. 16).

• The SMTP endpoint address. For a list of Amazon SES SMTP endpoints, see Connecting to an Amazon SES SMTP endpoint (p. 90).

• The Amazon SES SMTP interface port number, which depends on the connection method. For more information, see Connecting to an Amazon SES SMTP endpoint (p. 90).

After you get your SMTP credentials, you can connect to the Amazon SES SMTP endpoint and send email. This getting started tutorial shows you how to send email through the Amazon SES SMTP interface by using the following methods:

• Send an email by accessing the Amazon SES SMTP interface programmatically (p. 21)
• Configuring Your Existing Email Server or SMTP-Enabled Application to Send Email Through Amazon SES (p. 30)

For more information about the Amazon SES SMTP interface, see Using the Amazon SES SMTP interface to send email (p. 85).

**Send an email by accessing the Amazon SES SMTP interface programmatically**

You can access the Amazon SES SMTP interface by using an SMTP-enabled programming language. You provide the Amazon SES SMTP hostname and port number along with your SMTP credentials and then use the programming language’s generic SMTP functions to send the email.

**Note**
Amazon Elastic Compute Cloud (Amazon EC2) restricts email traffic over port 25 by default. To avoid timeouts when sending email through the SMTP endpoint from Amazon EC2, you can request that these restrictions be removed. For more information, see How do I remove the restriction on port 25 from my Amazon EC2 instance or AWS Lambda function? in the AWS Knowledge Center.

The code examples in this section use port 587 to avoid this issue.

**Topics in this section:**
• Send an email using SMTP with C# (p. 21)
• Send an email using SMTP with Java (p. 24)
• Send an email using SMTP with PHP (p. 27)

**Send an email using SMTP with C#**

The following procedure shows how to use Microsoft Visual Studio to create a C# console application that sends an email through Amazon SES. The procedures in this section apply to Visual Studio 2017, but the process of creating C# console applications is similar across Microsoft Visual Studio editions.

Before you perform the following procedure, complete the setup tasks described in Before you begin with Amazon SES (p. 19) and Send an email through Amazon SES using SMTP (p. 20).

**Important**
In this getting started tutorial, you send an email to yourself so that you can check to see if you received it. For further experimentation or load testing, use the Amazon SES mailbox simulator.
To send an email using the Amazon SES SMTP interface with C#

1. Create a console project in Visual Studio by performing the following steps:
   a. Open Microsoft Visual Studio.
   c. On the New Project window, in the left pane, expand Installed, expand Templates, and then expand Visual C#.
   e. On the menu at the top of the window, choose .NET Framework 4.5, as shown in the following image.

   ![New Project window with .NET Framework 4.5 selected](image)

   **Note**
   You can select a later version of the .NET Framework if necessary.

   g. In the Name field, type AmazonSESSample.
   h. Choose OK.

2. In your Visual Studio project, replace the entire contents of Program.cs with the following code:

```csharp
using System;
using System.Net;
using System.Net.Mail;

namespace AmazonSESSample
{
    class Program
    {
        static void Main(string[] args)
        {
            // Replace sender@example.com with your "From" address.
            // This address must be verified with Amazon SES.
            String FROM = "sender@example.com";
            String FROMNAME = "Sender Name";

            // Replace recipient@example.com with a "To" address. If your account
            // is still in the sandbox, this address must be verified.
            String TO = "recipient@amazon.com";

            // Replace smtp_username with your Amazon SES SMTP user name.
            String SMTP_USERNAME = "smtp_username";

            // Replace smtp_password with your Amazon SES SMTP password.
            String SMTP_PASSWORD = "smtp_password";

            // (Optional) the name of a configuration set to use for this message.
```
Send an email using SMTP

// If you comment out this line, you also need to remove or comment out
// the "X-SES-CONFIGURATION-SET" header below.
String CONFIGSET = "ConfigSet";

// If you're using Amazon SES in a region other than US West (Oregon),
// replace email-smtp.us-west-2.amazonaws.com with the Amazon SES SMTP
// endpoint in the appropriate AWS Region.
String HOST = "email-smtp.us-west-2.amazonaws.com";

// The port you will connect to on the Amazon SES SMTP endpoint. We
// are choosing port 587 because we will use STARTTLS to encrypt
// the connection.
int PORT = 587;

// The subject line of the email
String SUBJECT = "Amazon SES test (SMTP interface accessed using C#)";

// The body of the email
String BODY = "<h1>Amazon SES Test</h1>" +
"<p>This email was sent through the " +
"<a href='https://aws.amazon.com/ses'>Amazon SES</a> SMTP interface " +
"using the .NET System.Net.Mail library.</p>";

// Create and build a new MailMessage object
MailMessage message = new MailMessage();
message.IsBodyHtml = true;
message.From = new MailAddress(FROM, FROMNAME);
message.To.Add(new MailAddress(TO));
message.Subject = SUBJECT;
message.Body = BODY;
// Comment or delete the next line if you are not using a configuration set
message.Headers.Add("X-SES-CONFIGURATION-SET", CONFIGSET);

{
    // Pass SMTP credentials
    client.Credentials =
        new NetworkCredential(SMTP_USERNAME, SMTP_PASSWORD);

    // Enable SSL encryption
    client.EnableSsl = true;

    // Try to send the message. Show status in console.
    try
    {
        Console.WriteLine("Attempting to send email...");
        client.Send(message);
        Console.WriteLine("Email sent!");
    }
    catch (Exception ex)
    {
        Console.WriteLine("The email was not sent.");
        Console.WriteLine("Error message: " + ex.Message);
    }
}
}

3. In Program.cs, replace the following email addresses with your own values:
**Send an email using SMTP**

**Important**
The email addresses are case-sensitive. Make sure that the addresses are exactly the same as the ones you verified.

- **SENDER@EXAMPLE.COM**—Replace with your "From" email address. You must verify this address before you run this program. For more information, see Verifying identities in Amazon SES (p. 47).
- **RECIPIENT@EXAMPLE.COM**—Replace with your "To" email address. If your account is still in the sandbox, you must verify this address before you use it. For more information, see Moving out of the Amazon SES sandbox (p. 71).

4. In **Program.cs**, replace the following SMTP credentials with the values that you obtained in Obtaining your Amazon SES SMTP credentials (p. 86):

**Important**
Your SMTP credentials are different from your AWS credentials. For more information about credentials, see Types of Amazon SES credentials (p. 16).

- **YOUR_SMTP_USERNAME**—Replace with your SMTP username. Note that your SMTP username credential is a 20-character string of letters and numbers, not an intelligible name.
- **YOUR_SMTP_PASSWORD**—Replace with your SMTP password.

5. (Optional) If you want to use an Amazon SES SMTP endpoint in a Region other than US West (Oregon), change the value of the variable **HOST** to the endpoint you want to use. For a list of SMTP endpoint URLs for the AWS Regions where Amazon SES is available, see Amazon Simple Email Service (Amazon SES) in the AWS General Reference.

6. (Optional) If you want to use a configuration set when sending this email, change the value of the variable **CONFIGSET** to the name of the configuration set. For more information about configuration sets, see Using Amazon SES configuration sets (p. 249).

7. Save **Program.cs**.

8. To build the project, choose **Build** and then choose **Build Solution**.

9. To run the program, choose **Debug** and then choose **Start Debugging**.

10. Review the output. If the email was successfully sent, the console displays "Email sent!" Otherwise, it displays an error message.

11. Sign in to the email client of the recipient address. You will find the message that you sent.

**Send an email using SMTP with Java**

This example uses the Eclipse IDE and the JavaMail API to send email through Amazon SES using the SMTP interface.

Before you perform the following procedure, complete the setup tasks described in Before you begin with Amazon SES (p. 19) and Send an email through Amazon SES using SMTP (p. 20).

**Important**
In this getting started tutorial, you send an email to yourself so that you can check to see if you received it. For further experimentation or load testing, use the Amazon SES mailbox simulator. Emails that you send to the mailbox simulator do not count toward your sending quota or your bounce and complaint rates. For more information, see Testing email sending in Amazon SES (p. 181).

**To send an email using the Amazon SES SMTP interface with Java**

1. In a web browser, go to the JavaMail Github page. Under **Downloads**, choose **javax.mail.jar** to download the latest version of JavaMail.
Important
This tutorial requires JavaMail version 1.5 or later. These procedures were tested using
JavaMail version 1.6.1.

2. Create a project in Eclipse by performing the following steps:
   a. Start Eclipse.
   b. In Eclipse, choose File, choose New, and then choose Java Project.
   c. In the Create a Java Project dialog box, type a project name and then choose Next.
   d. In the Java Settings dialog box, choose the Libraries tab.
   e. Choose Add External JARs.
   f. Browse to the folder in which you downloaded JavaMail. Choose the file javax.mail.jar, and then choose Open.
   g. In the Java Settings dialog box, choose Finish.

3. In Eclipse, in the Package Explorer window, expand your project.

4. Under your project, right-click the src directory, choose New, and then choose Class.

5. In the New Java Class dialog box, in the Name field, type AmazonSESSample and then choose Finish.

6. Replace the entire contents of AmazonSESSample.java with the following code:

```java
import java.util.Properties;
import javax.mail.Message;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;

public class AmazonSESSample {

   // Replace sender@example.com with your "From" address.
   // This address must be verified.
   static final String FROM = "sender@example.com";
   static final String FROMNAME = "Sender Name";

   // Replace recipient@example.com with a "To" address. If your account
   // is still in the sandbox, this address must be verified.
   static final String TO = "recipient@example.com";

   // Replace smtp_username with your Amazon SES SMTP user name.
   static final String SMTP_USERNAME = "smtp_username";

   // Replace smtp_password with your Amazon SES SMTP password.
   static final String SMTP_PASSWORD = "smtp_password";

   // The name of the Configuration Set to use for this message.
   // If you comment out or remove this variable, you will also need to
   // comment out or remove the header below.
   static final String CONFIGSET = "ConfigSet";

   // Amazon SES SMTP host name. This example uses the US West (Oregon) region.
   // See https://docs.aws.amazon.com/ses/latest/DeveloperGuide/regions.html#region-endpoints
   // for more information.
   static final String HOST = "email-smtp.us-west-2.amazonaws.com";

   // The port you will connect to on the Amazon SES SMTP endpoint.
   static final int PORT = 587;

   // ...

   public static void main(String[] args) {
      Properties properties = System.getProperties();
      properties.put("mail.smtp.host", HOST);
      properties.put("mail.smtp.port", PORT);
      properties.put("mail.smtp.auth", "true");
      properties.put("mail.smtp.starttls.enable", "true");
      Session session = Session.getInstance(properties, new Authenticator() {
         protected PasswordAuthentication getPasswordAuthentication() {
            return new PasswordAuthentication(SMTP_USERNAME, SMTP_PASSWORD);
         }
      });
      MimeMessage message = new MimeMessage(session);
      message.setFrom(new InternetAddress(FROM, FROMNAME));
      message.setRecipients(Message.RecipientType.TO, InternetAddress.parse(TO));
      message.setSubject("Amazon Simple Email Service Classic API Test Message");
      message.setText("This message is sent with Amazon Simple Email Service Classic API.
      
      Testing body of message.
      
      Thank you,
      Amazon SES");
      Transport.send(message);
   }
}
```
static final String SUBJECT = "Amazon SES test (SMTP interface accessed using Java)";

static final String BODY = String.join(
    System.getProperty("line.separator"),
    "<h1>Amazon SES SMTP Email Test</h1>",
    "<p>This email was sent with Amazon SES using the ",
    "<a href='https://github.com/javaee/javamail'>Javamail Package</a>",
    ");

public static void main(String[] args) throws Exception {
    // Create a Properties object to contain connection configuration information.
    Properties props = System.getProperties();
    props.put("mail.transport.protocol", "smtp");
    props.put("mail.smtp.port", PORT);
    props.put("mail.smtp.starttls.enable", "true");
    props.put("mail.smtp.auth", "true");
    // Create a Session object to represent a mail session with the specified
    // properties.
    Session session = Session.getDefaultInstance(props);
    // Create a message with the specified information.
    MimeMessage msg = new MimeMessage(session);
    msg.setFrom(new InternetAddress(FROM,FROMNAME));
    msg.setRecipient(Message.RecipientType.TO, new InternetAddress(TO));
    msg.setSubject(SUBJECT);
    msg.setContent(BODY,"text/html");
    // Add a configuration set header. Comment or delete the
    // next line if you are not using a configuration set
    msg.setHeader("X-SES-CONFIGURATION-SET", CONFIGSET);
    // Create a transport.
    Transport transport = session.getTransport();
    // Send the message.
    try {
        System.out.println("Sending...");
        // Connect to Amazon SES using the SMTP username and password you specified
        above.
        transport.connect(HOST, SMTP_USERNAME, SMTP_PASSWORD);
        // Send the email.
        transport.sendMessage(msg, msg.getAllRecipients());
        System.out.println("Email sent!");
    } catch (Exception ex) {
        System.out.println("The email was not sent.");
        System.out.println("Error message: " + ex.getMessage());
    } finally
    {
        // Close and terminate the connection.
        transport.close();
    }
}

7. In AmazonSESSample.java, replace the following email addresses with your own values:
Send an email using SMTP

Important
The email addresses are case-sensitive. Make sure that the addresses are exactly the same as the ones you verified.

- SENDER@EXAMPLE.COM—Replace with your "From" email address. You must verify this address before you run this program. For more information, see Verifying identities in Amazon SES (p. 47).
- RECIPIENT@EXAMPLE.COM—Replace with your "To" email address. If your account is still in the sandbox, you must verify this address before you use it. For more information, see Moving out of the Amazon SES sandbox (p. 71).

8. In AmazonSESSample.java, replace the following SMTP credentials with the values that you obtained in Obtaining your Amazon SES SMTP credentials (p. 86):

   Important
   Your SMTP credentials are different from your AWS credentials. For more information about credentials, see Types of Amazon SES credentials (p. 16).

   - YOUR_SMTP_USERNAME—Replace with your SMTP username credential. Note that your SMTP username credential is a 20-character string of letters and numbers, not an intelligible name.
   - YOUR_SMTP_PASSWORD—Replace with your SMTP password.

9. (Optional) If you want to use an Amazon SES SMTP endpoint in an AWS Region other than US West (Oregon), change the value of the variable HOST to the endpoint you want to use. For a list of regions where Amazon SES is available, see Amazon Simple Email Service (Amazon SES) in the AWS General Reference.

10. (Optional) If you want to use a configuration set when sending this email, change the value of the variable CONFIGSET to the name of the configuration set. For more information about configuration sets, see Using Amazon SES configuration sets (p. 249).

11. Save AmazonSESSample.java.

12. To build the project, choose Project and then choose Build Project. (If this option is disabled, then you may have automatic building enabled.)

13. To start the program and send the email, choose Run and then choose Run again.

14. Review the output. If the email was successfully sent, the console displays "Email sent!" Otherwise, it displays an error message.

15. Sign into the email client of the recipient address. You will find the message that you sent.

Send an email using SMTP with PHP

This example uses the PHPMailer class to send email through Amazon SES using the SMTP interface.

Important
In this tutorial, you send an email to yourself so that you can check to see if you received it. For further experimentation or load testing, use the Amazon SES mailbox simulator. Emails that you send to the mailbox simulator do not count toward your sending quota or your bounce and complaint rates. For more information, see Testing email sending in Amazon SES (p. 181).

Prerequisites
Before you begin, perform the following tasks:

- Verify your email address with Amazon SES— Before you can send an email with Amazon SES, you must verify that you own the sender's email address. If your account is still in the Amazon SES sandbox, you must also verify the recipient email address. The easiest way to verify email addresses is by using the Amazon SES console. For more information, see Verifying email addresses in Amazon SES (p. 47).
- **Get your SMTP credentials**—You need an Amazon SES SMTP user name and password to access the Amazon SES SMTP interface. Your SMTP credentials are **not** the same as your AWS credentials. You can find your SMTP credentials by going to the SMTP Settings page of the Amazon SES console. For more information about SMTP credentials, see Obtaining your Amazon SES SMTP credentials (p. 86).

- **Install PHP**—PHP is available at [http://php.net/downloads.php](http://php.net/downloads.php). After you install PHP, add the path to PHP in your environment variables so that you can run PHP from any command prompt.

- **Install the Composer dependency manager**—The Composer dependency manager will enable you to download and install the PHPMailer class and its dependencies. To install Composer, follow the installation instructions at [https://getcomposer.org/download](https://getcomposer.org/download).

- **Install the PHPMailer class**—After you install Composer, run the following command to install PHPMailer:

  ```bash
  path/to/composer require phpmailer/phpmailer
  ```

  In the preceding command, replace `path/to/` with the path where you installed Composer.

**Procedure**

The following procedure shows how to send an email through Amazon SES with PHP.

**To send an email using the Amazon SES SMTP interface with PHP**

1. Create a file named `amazon-ses-smtp-sample.php`. Open the file with a text editor and paste in the following code:

   ```php
   <?php
   // Import PHPMailer classes into the global namespace
   // These must be at the top of your script, not inside a function
   use PHPMailer\PHPMailer\PHPMailer;
   use PHPMailer\PHPMailer\Exception;
   // If necessary, modify the path in the require statement below to refer to the
   // location of your Composer autoload.php file.
   require 'vendor/autoload.php';

   // Replace sender@example.com with your "From" address.
   // This address must be verified with Amazon SES.
   $sender = 'sender@example.com';
   $senderName = 'Sender Name';

   // Replace recipient@example.com with a "To" address. If your account
   // is still in the sandbox, this address must be verified.
   $recipient = 'recipient@example.com';

   // Replace smtp_username with your Amazon SES SMTP user name.
   $usernameSmtp = 'smtp_username';

   // Replace smtp_password with your Amazon SES SMTP password.
   $passwordSmtp = 'smtp_password';

   // Specify a configuration set. If you do not want to use a configuration
   // set, comment or remove the next line.
   $configurationSet = 'ConfigSet';

   // If you're using Amazon SES in a region other than US West (Oregon),
   // replace email-smtp.us-west-2.amazonaws.com with the Amazon SES SMTP
   // endpoint in the appropriate region.
   $host = 'email-smtp.us-west-2.amazonaws.com';
   $port = 587;
   ```
In `amazon-ses-smtp-sample.php`, replace the following with your own values:

- **sender@example.com**—Replace with an email address that you have verified with Amazon SES. For more information, see Verifying identities (p. 47). Email addresses in Amazon SES are case-sensitive. Make sure that the address you enter is exactly the same as the one you verified.

- **recipient@example.com**—Replace with the address of the recipient. If your account is still in the sandbox, you must verify this address before you use it. For more information, see Moving out of the Amazon SES sandbox (p. 71). Make sure that the address you enter is exactly the same as the one you verified.

- **smtp_username**—Replace with your SMTP user name credential, which you obtained from the SMTP Settings page of the Amazon SES console. This is **not** the same as your AWS access key ID. Note that your SMTP user name credential is a 20-character string of letters and numbers, not an intelligible name.
• **smtp_password**—Replace with your SMTP password, which you obtained from the SMTP Settings page of the Amazon SES console. This is **not** the same as your AWS secret access key.

• **(Optional) ConfigSet**—If you want to use a configuration set when sending this email, replace this value with the name of the configuration set. For more information about configuration sets, see Using Amazon SES configuration sets (p. 249).

• **(Optional) email-smtp.us-west-2.amazonaws.com**—If you want to use an Amazon SES SMTP endpoint in a Region other than US West (Oregon), replace this with the Amazon SES SMTP endpoint in the Region you want to use. For a list of SMTP endpoint URLs for the AWS Regions where Amazon SES is available, see Amazon Simple Email Service (Amazon SES) in the AWS General Reference.


4. To run the program, open a command prompt in the same directory as `amazon-ses-smtp-sample.php`, and then type `php amazon-ses-smtp-sample.php`.

5. Review the output. If the email was successfully sent, the console displays "Email sent!" Otherwise, it displays an error message.

6. Sign in to the email client of the recipient address. You will find the message that you sent.

### Configuring Your Existing Email Server or SMTP-Enabled Application to Send Email Through Amazon SES

You can configure your mail server, email client, or email sending software package to send messages through Amazon SES without any programming.

First, read Send an email through Amazon SES using SMTP (p. 20). Then review one of the following topics, which show you how to configure a mail server to forward mail to Amazon SES:

- Configuring Postfix (p. 94)
- Integrating Amazon SES with Sendmail (p. 97)
- Integrating Amazon SES with Exim (p. 101)

### Send an email through Amazon SES using an AWS SDK

To send an email using the Amazon SES API, you can use the Query interface directly, or you can use an AWS SDK to handle low-level details such as assembling and parsing HTTP requests and responses.

Before you send email using an AWS SDK, review the instructions in Before you begin with Amazon SES (p. 19). In order to complete the tutorials in this section, you also need to:

- **Download an AWS SDK**—Download and install an AWS SDK. For more information, see Downloading an AWS SDK (p. 65).

- **Get your AWS credentials**—To access Amazon SES programmatically, you need your AWS access keys. For more information, see Getting your AWS access keys (p. 65).

- **Create a shared credentials file**—Follow the procedures in Create a shared credentials file (p. 31) to create the shared credentials file.

When you have completed the prerequisites listed above, see Send an email through Amazon SES programmatically using an AWS SDK (p. 31).
Create a shared credentials file

The following procedure shows how to create a shared credentials file in your home directory. For the SDK sample code to function properly, you must create this file.

1. In a text editor, create a new file. In the file, paste the following code:

```
[default]
aws_access_key_id = YOUR_AWS_ACCESS_KEY_ID
aws_secret_access_key = YOUR_AWS_SECRET_ACCESS_KEY
```

2. In the text file you just created, replace `YOUR_AWS_ACCESS_KEY_ID` with your unique AWS access key ID, and replace `YOUR_AWS_SECRET_ACCESS_KEY` with your unique AWS secret access key.

3. Save the file. The following table shows the correct location and file name for your operating system.

<table>
<thead>
<tr>
<th>If you're using...</th>
<th>Save the file as...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows</td>
<td>C:\Users&lt;yourUserName&gt;.aws\credentials</td>
</tr>
<tr>
<td>Linux, macOS, or Unix</td>
<td>~/.aws/credentials</td>
</tr>
</tbody>
</table>

**Important**

Don’t include a file extension when saving the credentials file.

Send an email through Amazon SES programmatically using an AWS SDK

You can use an AWS SDK to send email through Amazon SES. AWS SDKs are available for several programming languages. For more information, see Tools for Amazon Web Services.

**Note**

If you have not already done so, complete the prerequisites listed in Send an email through Amazon SES using an AWS SDK (p. 30) before you attempt to complete the tutorials in this section.

You can find additional code examples in Amazon SES code examples (p. 416).

Topics in this section:

- Send an email using the AWS SDK for .NET (p. 31)
- Send an email using the AWS SDK for Java (p. 34)
- Send an email using the AWS SDK for PHP (p. 37)
- Send an email using the AWS SDK for Ruby (p. 40)
- Send an email using the AWS SDK for Python (Boto) (p. 42)

Send an email using the AWS SDK for .NET

The following procedure shows you how to send an email through Amazon SES using Visual Studio and the AWS SDK for .NET.

This solution was tested using the following components:

- Microsoft Visual Studio Community 2017, version 15.4.0.
• Microsoft .NET Framework version 4.6.1.
• The AWSSDK.Core package (version 3.3.19), installed using NuGet.
• The AWSSDK.SimpleEmail package (version 3.3.6.1), installed using NuGet.

Note
In this getting started tutorial, you send an email to yourself so that you can check to see if you received it. For further experimentation or load testing, use the Amazon SES mailbox simulator. Emails that you send to the mailbox simulator do not count toward your sending quota or your bounce and complaint rates. For more information, see Testing email sending in Amazon SES (p. 181).

Prerequisites
Before you begin, perform the following tasks:

• Verify your email address with Amazon SES—Before you can send an email with Amazon SES, you must verify that you own the sender’s email address. If your account is still in the Amazon SES sandbox, you must also verify the recipient email address. The easiest way to verify email addresses is by using the Amazon SES console. For more information, see Verifying email addresses in Amazon SES (p. 47).
• Get your AWS credentials—you need an AWS access key ID and AWS secret access key to access Amazon SES using an SDK. You can find your credentials by using the Security Credentials page of the AWS Management Console. For more information about credentials, see Types of Amazon SES credentials (p. 16).
• Install Visual Studio—Visual Studio is available at https://www.visualstudio.com/.
• Create a shared credentials file—for the sample code in this section to function properly, you must create a shared credentials file. For more information, see Create a shared credentials file (p. 31).

Procedure
The following procedure shows how to send an email through Amazon SES using the AWS SDK for .NET.

To send an email using the AWS SDK for .NET

1. Create a new project by performing the following steps:
   c. On the New Project window, in the panel on the left, expand Installed, and then expand Visual C#.
   d. In the panel on the right, choose Console App (.NET Framework).
   e. For Name, type AmazonSESSample, and then choose OK.
2. Use NuGet to include the Amazon SES packages in your solution by completing the following steps:
   a. In the Solution Explorer pane, right-click your project, and then choose Manage NuGet Packages.
   b. On the NuGet: AmazonSESSample tab, choose Browse.
   c. In the search box, type AWSSDK.SimpleEmail.
   d. Choose the AWSSDK.SimpleEmail package, and then choose Install.
   e. On the Preview Changes window, choose OK.
3. On the Program.cs tab, paste the following code:

```csharp
using Amazon;
```
using System;
using System.Collections.Generic;
using Amazon.SimpleEmail;
using Amazon.SimpleEmail.Model;

namespace AmazonSESSample
{
    class Program
    {
        // Replace sender@example.com with your "From" address.
        // This address must be verified with Amazon SES.
        static readonly string senderAddress = "sender@example.com";

        // Replace recipient@example.com with a "To" address. If your account
        // is still in the sandbox, this address must be verified.
        static readonly string receiverAddress = "recipient@example.com";

        // The configuration set to use for this email. If you do not want to use a
        // configuration set, comment out the following property and the
        // ConfigurationSetName = configSet argument below.
        static readonly string configSet = "ConfigSet";

        // The subject line for the email.
        static readonly string subject = "Amazon SES test (AWS SDK for .NET)";

        // The email body for recipients with non-HTML email clients.
        static readonly string textBody = "Amazon SES Test (.NET)\r\n" + "This email was sent through Amazon SES " + "using the AWS SDK for .NET."

        // The HTML body of the email.
        static readonly string htmlBody = "<html>
<head></head>
<body>
<h1>Amazon SES Test (AWS SDK for .NET)</h1>
<p>This email was sent with
<a href='https://aws.amazon.com/ses/'>Amazon SES</a> using the
<a href='https://aws.amazon.com/sdk-for-net/'>AWS SDK for .NET</a>.</p>
</body>
</html>"

        static void Main(string[] args)
        {
            // Replace USWest2 with the AWS Region you're using for Amazon SES.
            // Acceptable values are EUWest1, USEast1, and USWest2.
            using (var client = new
            AmazonSimpleEmailServiceClient(RegionEndpoint.USWest2))
            {
                var sendRequest = new SendEmailRequest
                {
                    Source = senderAddress,
                    Destination = new Destination
                    {
                        ToAddresses = new List<string> { receiverAddress }
                    },
                    Message = new Message
                    {
                        Subject = new Content(subject),
                        Body = new Body
                        {
                            Html = new Content
                            {
                                Charset = "UTF-8",
                                Data = htmlBody
                            }
                        }
                    }
                };
            }
        }
    }
}
Send an email using an AWS SDK

```csharp
// Text = new Content
{       // If you are not using a configuration set, comment
    Charset = "UTF-8", // or remove the following line
    Data = textBody
}

// ConfigurationSetName = configSet
try
{
    Console.WriteLine("Sending email using Amazon SES... ");
    var response = client.SendEmail(sendRequest);
    Console.WriteLine("The email was sent successfully. ");
}
catch (Exception ex)
{
    Console.WriteLine("The email was not sent. ");
    Console.WriteLine("Error message: " + ex.Message);
}

Console.WriteLine("Press any key to continue... ");
Console.ReadKey();

4. In the code editor, do the following:
   - Replace sender@example.com with the "From:" email address. This address must be verified. For more information, see the section called “Verifying identities” (p. 47).
   - Replace recipient@example.com with the "To:" address. If your account is still in the sandbox, this address must also be verified.
   - Replace ConfigSet with the name of the configuration set to use when sending this email.
   - Replace USWest2 with the name of the AWS Region endpoint you use to send email using Amazon SES. For a list of regions where Amazon SES is available, see Amazon Simple Email Service (Amazon SES) in the AWS General Reference.

When you finish, save Program.cs.

5. Build and run the application by completing the following steps:
   a. On the Build menu, choose Build Solution.

6. Review the output of the console. If the email was successfully sent, the console displays "The email was sent successfully."

7. If the email was successfully sent, sign in to the email client of the recipient address. You will find the message that you sent.

Send an email using the AWS SDK for Java

The following procedure shows you how to use Eclipse IDE for Java EE Developers and AWS Toolkit for Eclipse to create an AWS SDK project and modify the Java code to send an email through Amazon SES.
Important
In this getting started tutorial, you send an email to yourself so that you can check to see if you received it. For further experimentation or load testing, use the Amazon SES mailbox simulator. Emails that you send to the mailbox simulator do not count toward your sending quota or your bounce and complaint rates. For more information, see Testing email sending in Amazon SES (p. 181).

Prerequisites

Before you begin, perform the following tasks:

- **Verify your email address with Amazon SES**—Before you can send an email with Amazon SES, you must verify that you own the sender's email address. If your account is still in the Amazon SES sandbox, you must also verify the recipient email address. The easiest way to verify email addresses is by using the Amazon SES console. For more information, see Verifying email addresses in Amazon SES (p. 47).
- **Get your AWS credentials**—You need an AWS access key ID and AWS secret access key to access Amazon SES using an SDK. You can find your credentials by using the Security Credentials page in the AWS Management Console. For more information about credentials, see Types of Amazon SES credentials (p. 16).
- **Install Eclipse**—Eclipse is available at https://www.eclipse.org/downloads. The code in this tutorial was tested using Eclipse Neon.3 (version 4.6.3), running version 1.8 of the Java Runtime Environment.
- **Install the AWS Toolkit for Eclipse**—Instructions for adding the AWS Toolkit for Eclipse to your Eclipse installation are available at https://aws.amazon.com/eclipse. The code in this tutorial was tested using version 2.3.1 of the AWS Toolkit for Eclipse.
- **Create a shared credentials file**—For the sample code in this section to function properly, you must create a shared credentials file. For more information, see Create a shared credentials file (p. 31).

Procedure

The following procedure shows how to send an email through Amazon SES using the AWS SDK for Java.

**To send an email using the AWS SDK for Java**

1. Create an AWS Java Project in Eclipse by performing the following steps:
   a. Start Eclipse.
   b. On the **File** menu, choose **New**, and then choose **Other**. On the **New** window, expand the **AWS** folder, and then choose **AWS Java Project**.
   c. In the **New AWS Java Project** dialog box, do the following:
      i. For **Project name**, type a project name.
      ii. Under **AWS SDK for Java Samples**, select **Amazon Simple Email Service JavaMail Sample**.
      iii. Choose **Finish**.
2. In Eclipse, in the **Package Explorer** pane, expand your project.
3. Under your project, expand the **src/main/java** folder, expand the **com.amazon.aws.samples** folder, and then double-click **AmazonSESSample.java**.
4. Replace the entire contents of **AmazonSESSample.java** with the following code:

```java
package com.amazonaws.samples;
import java.io.IOException;
import com.amazonaws.regions.Regions;
```
public class AmazonSESSample {

    // Replace sender@example.com with your "From" address.
    // This address must be verified with Amazon SES.
    static final String FROM = "sender@example.com";

    // Replace recipient@example.com with a "To" address. If your account
    // is still in the sandbox, this address must be verified.
    static final String TO = "recipient@example.com";

    // The configuration set to use for this email. If you do not want to use a
    // configuration set, comment the following variable and the
    // .withConfigurationSetName(CONFIGSET); argument below.
    static final String CONFIGSET = "ConfigSet";

    // The subject line for the email.
    static final String SUBJECT = "Amazon SES test (AWS SDK for Java)";

    // The HTML body for the email.
    static final String HTMLBODY = "<h1>Amazon SES test (AWS SDK for Java)</h1>
    + "This email was sent with <a href='https://aws.amazon.com/ses/'>Amazon SES</a> using the <a href='https://aws.amazon.com/sdk-for-java/'>AWS SDK for Java</a>";

    // The email body for recipients with non-HTML email clients.
    static final String TEXTBODY = "This email was sent through Amazon SES " + "using the AWS SDK for Java.";

    public static void main(String[] args) throws IOException {
        try {
            AmazonSimpleEmailService client = AmazonSimpleEmailServiceClientBuilder.standard()
                .withRegion(Regions.US_WEST_2).build();
            SendEmailRequest request = new SendEmailRequest()
                .withDestination(new Destination().withToAddresses(TO))
                .withMessage(new Message()
                    .withBody(new Body()
                        .withHtml(new Content()
                            .withCharset("UTF-8").withData(HTMLBODY)
                        )
                        .withText(new Content()
                            .withCharset("UTF-8").withData(TEXTBODY)
                        )
                        .withSubject(new Content()
                            .withCharset("UTF-8").withData(SUBJECT)
                        )
                    )
                )
                .withSource(FROM);
            client.sendEmail(request);
            System.out.println("Email sent!");
        } catch (Exception ex) {
            System.out.println("The email was not sent. Error message: " + ex.getMessage());
        }
    }
}
5. In AmazonSESSample.java, replace the following with your own values:

   **Important**
   The email addresses are case-sensitive. Make sure that the addresses are exactly the same as the ones you verified.

   - SENDER@EXAMPLE.COM—Replace with your "From" email address. You must verify this address before you run this program. For more information, see Verifying identities in Amazon SES (p. 47).
   - RECIPIENT@EXAMPLE.COM—Replace with your "To" email address. If your account is still in the sandbox, you must verify this address before you use it. For more information, see Moving out of the Amazon SES sandbox (p. 71).
   - (Optional) us-west-2—If you want to use Amazon SES in a region other than US West (Oregon), replace this with the region you want to use. For a list of regions where Amazon SES is available, see Amazon Simple Email Service (Amazon SES) in the AWS General Reference.

6. Save AmazonSESSample.java.

7. To build the project, choose Project and then choose Build Project.

   **Note**
   If this option is disabled, automatic building may be enabled; if so, skip this step.

8. To start the program and send the email, choose Run and then choose Run again.

9. Review the output of the console pane in Eclipse. If the email was successfully sent, the console displays "Email sent!" Otherwise, it displays an error message.

10. If the email was successfully sent, sign in to the email client of the recipient address. You will find the message that you sent.

### Send an email using the AWS SDK for PHP

This topic shows how to use the AWS SDK for PHP to send an email through Amazon SES.

   **Important**
   In this tutorial, you send an email to yourself so that you can check to see if you received it. For further experimentation or load testing, use the Amazon SES mailbox simulator. Emails that you send to the mailbox simulator do not count toward your sending quota or your bounce and complaint rates. For more information, see Testing email sending in Amazon SES (p. 181).

**Prerequisites**

Before you begin, perform the following tasks:

- **Verify your email address with Amazon SES**—Before you can send an email with Amazon SES, you must verify that you own the sender's email address. If your account is still in the Amazon SES sandbox, you must also verify the recipient email address. The easiest way to verify email addresses is by using the Amazon SES console. For more information, see Verifying email addresses in Amazon SES (p. 47).
- **Get your AWS credentials**—You need an AWS access key ID and AWS secret access key to access Amazon SES using an SDK. You can find your credentials by using the Security Credentials page of the AWS Management Console. For more information about credentials, see Types of Amazon SES credentials (p. 16).
- **Install PHP**—PHP is available at http://php.net/downloads.php. This tutorial requires PHP version 5.5 or higher. After you install PHP, add the path to PHP in your environment variables so that you can run PHP from any command prompt. The code in this tutorial was tested using PHP 7.2.7.
- **Install the AWS SDK for PHP version 3**—For download and installation instructions, see the AWS SDK for PHP documentation. The code in this tutorial was tested using version 3.64.13 of the SDK.
• **Create a shared credentials file**—For the sample code in this section to function properly, you must create a shared credentials file. For more information, see Create a shared credentials file (p. 31).

**Procedure**

The following procedure shows how to send an email through Amazon SES using the AWS SDK for PHP.

**To send an email through Amazon SES using the AWS SDK for PHP**

1. In a text editor, create a file named `amazon-ses-sample.php`. Paste the following code:

   ```php
   <?php
   // If necessary, modify the path in the require statement below to refer to the
   // location of your Composer autoload.php file.
   require 'vendor/autoload.php';

   use Aws\Ses\SesClient;
   use Aws\Exception\AwsException;

   // Create an SesClient. Change the value of the region parameter if you're
   // using an AWS Region other than US West (Oregon). Change the value of the
   // profile parameter if you want to use a profile in your credentials file
   // other than the default.
   $SesClient = new SesClient([  
   'profile' => 'default',  
   'version' => '2010-12-01',  
   'region'  => 'us-west-2'  
   ]);  

   // Replace sender@example.com with your "From" address.
   // This address must be verified with Amazon SES.
   $sender_email = 'sender@example.com';

   // Replace these sample addresses with the addresses of your recipients. If
   // your account is still in the sandbox, these addresses must be verified.
   $recipient_emails = ['recipient1@example.com','recipient2@example.com'];

   // Specify a configuration set. If you do not want to use a configuration
   // set, comment the following variable, and the
   // 'ConfigurationSetName' => $configuration_set argument below.
   $configuration_set = 'ConfigSet';

   $subject = 'Amazon SES test (AWS SDK for PHP)';
   $plaintext_body = 'This email was sent with Amazon SES using the AWS SDK for PHP.';
   $html_body = '<h1>AWS Amazon Simple Email Service Test Email</h1>
   <p>This email was sent with <a href="https://aws.amazon.com/ses/">Amazon SES</a> using the <a href="https://aws.amazon.com/sdk-for-php/">AWS SDK for PHP</a>.</p>
   
   Amazon SES is a service that lets you send e-mail messages to your customers.
   
   This email was sent with Amazon SES using the AWS SDK for PHP.
   ';
   $char_set = 'UTF-8';

   try {
   $result = $SesClient->sendEmail([  
   'Destination' => [  
   'ToAddresses' => $recipient_emails,
   ],  
   'ReplyToAddresses' => [$sender_email],
   'Source' => $sender_email,
   'Message' => [  
   'Body' => [  
   'Html' => [  
   'Charset' => $char_set,
   ]
   ]
   ]
   ]
   )
   
   } catch (AwsException $e) {
   // handle exception
   echo $e->getMessage();
   }
   
   //... more code...
   ```
In `amazon-ses-sample.php`, replace the following with your own values:

- **path_to_sdk_inclusion**—Replace with the path required to include the AWS SDK for PHP in the program. For more information, see the [AWS SDK for PHP documentation](https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/).

- **sender@example.com**—Replace with an email address that you have verified with Amazon SES. For more information, see [Verifying identities](https://docs.aws.amazon.com/AmazonSes/latestDeveloperGuide/verifying-identities.html). Email addresses in Amazon SES are case-sensitive. Make sure that the address you enter is exactly the same as the one you verified.

- **recipient1@example.com, recipient2@example.com**—Replace with the addresses of your recipients. If your account is still in the sandbox, your recipients' addresses must also be verified. For more information, see [Moving out of the Amazon SES sandbox](https://docs.aws.amazon.com/AmazonSes/latestDeveloperGuide/moving-out-sandbox.html). Make sure that the address you enter is exactly the same as the one you verified.

- **(Optional) ConfigSet**—If you want to use a configuration set when sending this email, replace this value with the name of the configuration set. For more information about configuration sets, see [Using Amazon SES configuration sets](https://docs.aws.amazon.com/AmazonSes/latestDeveloperGuide/configuration-sets.html).

- **(Optional) us-west-2**—If you want to use Amazon SES in a region other than US West (Oregon), replace this with the region you want to use. For a list of regions where Amazon SES is available, see [Amazon Simple Email Service (Amazon SES)](https://docs.aws.amazon.com/AmazonSes/latestDeveloperGuide/) in the **AWS General Reference**.


4. To run the program, open a command prompt in the same directory as `amazon-ses-sample.php`, and then type the following command:

```
$ php amazon-ses-sample.php
```

5. Review the output. If the email was successfully sent, the console displays "Email sent!" Otherwise, it displays an error message.

**Note**

If you encounter a "cURL error 60: SSL certificate problem" error when you run the program, download the latest CA bundle as described in the [AWS SDK for PHP documentation](https://docs.aws.amazon.com/sdk-for-php/v3/developer-guide/). Then, in `amazon-ses-sample.php`, add the following lines to the `SesClient::factory` array, replace `path_of_certs` with the path to the CA bundle you downloaded, and re-run the program.

```php
'http' => [
`
Send an email using an AWS SDK

This topic shows how to use the AWS SDK for Ruby to send an email through Amazon SES.

Important
In this tutorial, you send an email to yourself so that you can check to see if you received it. For further experimentation or load testing, use the Amazon SES mailbox simulator. Emails that you send to the mailbox simulator do not count toward your sending quota or your bounce and complaint rates. For more information, see Testing email sending in Amazon SES (p. 181).

Prerequisites

Before you begin, perform the following tasks:

• Verify your email address with Amazon SES—Before you can send an email with Amazon SES, you must verify that you own the sender's email address. If your account is still in the Amazon SES sandbox, you must also verify the recipient email address. The easiest way to verify email addresses is by using the Amazon SES console. For more information, see Verifying email addresses in Amazon SES (p. 47).

• Get your AWS credentials—You need an AWS access key ID and AWS secret access key to access Amazon SES using an SDK. You can find your credentials by using the Security Credentials page of the AWS Management Console. For more information about credentials, see Types of Amazon SES credentials (p. 16).

• Install Ruby—Ruby is available at https://www.ruby-lang.org/en/downloads/. The code in this tutorial was tested using Ruby 1.9.3. After you install Ruby, add the path to Ruby in your environment variables so that you can run Ruby from any command prompt.

• Install the AWS SDK for Ruby—For download and installation instructions, see Installing the AWS SDK for Ruby in the AWS SDK for Ruby Developer Guide. The sample code in this tutorial was tested using version 2.9.36 of the AWS SDK for Ruby.

• Create a shared credentials file—For the sample code in this section to function properly, you must create a shared credentials file. For more information, see Create a shared credentials file (p. 31).

Procedure

The following procedure shows how to send an email through Amazon SES using the AWS SDK for Ruby.

To send an email through Amazon SES using the AWS SDK for Ruby

1. In a text editor, create a file named amazon-ses-sample.rb. Paste the following code into the file:

```ruby
require 'aws-sdk'

# Replace sender@example.com with your "From" address.
# This address must be verified with Amazon SES.
sender = "sender@example.com"

# Replace recipient@example.com with a "To" address. If your account
# is still in the sandbox, this address must be verified.
recipient = "recipient@example.com"

# Specify a configuration set. If you do not want to use a configuration
# set, comment the following variable and the
```
# configuration_set_name: configsetname argument below.

configsetname = "ConfigSet"

# Replace us-west-2 with the AWS Region you're using for Amazon SES.

awsregion = "us-west-2"

# The subject line for the email.

subject = "Amazon SES test (AWS SDK for Ruby)"

# The HTML body of the email.

htmlbody =
  '<h1>Amazon SES test (AWS SDK for Ruby)</h1>
  <p>This email was sent with <a href="https://aws.amazon.com/ses/">Amazon SES</a> using the <a href="https://aws.amazon.com/sdk-for-ruby/">AWS SDK for Ruby</a>.
  
  AWS SDK for Ruby</a>.

# The email body for recipients with non-HTML email clients.

textbody = "This email was sent with Amazon SES using the AWS SDK for Ruby."

# Specify the text encoding scheme.

encoding = "UTF-8"

# Create a new SES resource and specify a region

ses = Aws::SES::Client.new(region: awsregion)

# Try to send the email.

begin

  # Provide the contents of the email.
  resp = ses.send_email({
    destination: {
      to_addresses: [
        recipient,
      ],
    },
    message: {
      body: {
        html: {
          charset: encoding,
          data: htmlbody,
        },
        text: {
          charset: encoding,
          data: textbody,
        },
      },
      subject: {
        charset: encoding,
        data: subject,
      },
    },
    source: sender,
    # Comment or remove the following line if you are not using
    # a configuration set
    configuration_set_name: configsetname,
  })
  puts "Email sent!"

  # If something goes wrong, display an error message.
  rescue Aws::SES::Errors::ServiceError => error
    puts "Email not sent. Error message: #{error}"
end

2. In amazon-ses-sample.rb, replace the following with your own values:
Send an email using an AWS SDK

- **sender@example.com**—Replace with an email address that you have verified with Amazon SES. For more information, see Verifying identities (p. 47). Email addresses in Amazon SES are case-sensitive. Make sure that the address you enter is exactly the same as the one you verified.

- **recipient@example.com**—Replace with the address of the recipient. If your account is still in the sandbox, you must verify this address before you use it. For more information, see Moving out of the Amazon SES sandbox (p. 71). Make sure that the address you enter is exactly the same as the one you verified.

- **(Optional) us-west-2**—If you want to use Amazon SES in a region other than US West (Oregon), replace this with the region you want to use. For a list of regions where Amazon SES is available, see Amazon Simple Email Service (Amazon SES) in the AWS General Reference.

4. To run the program, open a command prompt in the same directory as `amazon-ses-sample.rb`, and type `ruby amazon-ses-sample.rb`
5. Review the output. If the email was successfully sent, the console displays "Email sent!" Otherwise, it displays an error message.
6. Sign in to the email client of the recipient address. You will find the message that you sent.

**Send an email using the AWS SDK for Python (Boto)**

This topic shows how to use the AWS SDK for Python (Boto) to send an email through Amazon SES.

**Important**
In this tutorial, you send an email to yourself so that you can check to see if you received it. For further experimentation or load testing, use the Amazon SES mailbox simulator. Emails that you send to the mailbox simulator do not count toward your sending quota or your bounce and complaint rates. For more information, see Testing email sending in Amazon SES (p. 181).

**Prerequisites**
Before you begin, perform the following tasks:

- **Verify your email address with Amazon SES**—Before you can send an email with Amazon SES, you must verify that you own the sender's email address. If your account is still in the Amazon SES sandbox, you must also verify the recipient email address. The easiest way to verify email addresses is by using the Amazon SES console. For more information, see Verifying email addresses in Amazon SES (p. 47).

- **Get your AWS credentials**—You need an AWS access key ID and AWS secret access key to access Amazon SES using an SDK. You can find your credentials by using the Security Credentials page of the AWS Management Console. For more information about credentials, see Types of Amazon SES credentials (p. 16).

- **Install Python**—Python is available at https://www.python.org/downloads/. The code in this tutorial was tested using Python 2.7.6 and Python 3.6.1. After you install Python, add the path to Python in your environment variables so that you can run Python from any command prompt.

- **Install the AWS SDK for Python (Boto)**—For download and installation instructions, see the AWS SDK for Python (Boto) documentation. The sample code in this tutorial was tested using version 1.4.4 of the SDK for Python.

- **Create a shared credentials file**—For the sample code in this section to function properly, you must create a shared credentials file. For more information, see Create a shared credentials file (p. 31).

**Procedure**

The following procedure shows how to send an email through Amazon SES using the SDK for Python.
To send an email through Amazon SES using the SDK for Python

1. In a text editor, create a file named `amazon-ses-sample.py`. Paste the following code into the file:

```python
import boto3
from botocore.exceptions import ClientError

# Replace sender@example.com with your "From" address.
# This address must be verified with Amazon SES.
SENDER = "Sender Name <sender@example.com>"

# Replace recipient@example.com with a "To" address. If your account
# is still in the sandbox, this address must be verified.
RECIPIENT = "recipient@example.com"

# Specify a configuration set. If you do not want to use a configuration
# set, comment the following variable, and the
# ConfigurationSetName=CONFIGURATION_SET argument below.
CONFIGURATION_SET = "ConfigSet"

# If necessary, replace us-west-2 with the AWS Region you're using for Amazon SES.
AWS_REGION = "us-west-2"

# The subject line for the email.
SUBJECT = "Amazon SES Test (SDK for Python)"

# The email body for recipients with non-HTML email clients.
BODY_TEXT = "Amazon SES Test (Python)\r\n" + "This email was sent with Amazon SES using the " + "AWS SDK for Python (Boto)."

# The HTML body of the email.
BODY_HTML = "<html>" + "<body>" + "<h1>Amazon SES Test (SDK for Python)</h1>" + "<p>This email was sent with Amazon SES using the " + "AWS SDK for Python (Boto).</p>" + "</body>" + "</html>"

# The character encoding for the email.
CHARSET = "UTF-8"

# Create a new SES resource and specify a region.
client = boto3.client('ses', region_name=AWS_REGION)

try:
    #Provide the contents of the email.
    response = client.send_email(
        Destination={
            'ToAddresses': [RECIPIENT,],
        },
        Message={
            'Body': {
                'Html': {
                    'Charset': CHARSET,
                    'Data': BODY_HTML,
                },
            },
        },
    )
```

```diff
```
Migrate to Amazon SES from another solution

This topic provides an overview of the steps that you have to take if you want to move your email-sending solution to Amazon SES from a solution that's hosted on-premises, or from one hosted on an Amazon EC2 instance.

Topics in this section:
- Verify your domain (p. 45)
- Request production access (p. 45)
- Configure domain authentication systems (p. 45)
- Generate your SMTP credentials (p. 45)
- Connect to an SMTP endpoint (p. 45)
Verify your domain

Before you can use Amazon SES to send email, you have to verify the identities that you plan to send email from. In Amazon SES, an identity can be an email address or an entire domain. When you verify a domain, you can use Amazon SES to send email from any address on that domain. For more information about verifying a domain, see Verifying domains in Amazon SES (p. 59).

Request production access

When you first start using Amazon SES, your account is in a sandbox environment. While your account is in the sandbox, you can only send email to addresses that you've verified. Additionally, there are restrictions on the number of messages that you can send per day, and the number that you can send per second. For more information about requesting production access, see Moving out of the Amazon SES sandbox (p. 71).

Configure domain authentication systems

You can configure your domain to use authentication systems such as DKIM and SPF. This step is technically optional. However, by setting up either DKIM or SPF (or both) for your domain, you can improve the deliverability of your emails, and increase the amount of trust that your customers have in you. For more information about setting up SPF, see Authenticating Email with SPF in Amazon SES (p. 129). For more information about setting up DKIM, see Authenticating Email with DKIM in Amazon SES (p. 130).

Generate your SMTP credentials

If you plan to send email using an application that uses SMTP, you have to generate SMTP credentials. Your SMTP credentials are different from your regular AWS credentials. These credentials are also unique in each AWS Region. For more information about generating your SMTP credentials, see Obtaining your Amazon SES SMTP credentials (p. 86).

Connect to an SMTP endpoint

If you use a message transfer agent such as postfix or sendmail, you have to update the configuration for that application to refer to an Amazon SES SMTP endpoint. For a complete list of SMTP endpoints, see Connecting to an Amazon SES SMTP endpoint (p. 90). Note that the SMTP credentials that you created in the previous step are associated with a specific AWS Region. You have to connect to the SMTP endpoint in the region that you created the SMTP credentials in.

Next steps

At this point, you're ready to start sending email using Amazon SES. However, there are a few optional steps that you can take.

- You can create configuration sets, which are sets of rules that are applied to the emails that you send. For example, you can use configuration sets to specify where notifications are sent when an email is delivered, when a recipient opens a message or clicks a link in it, when an email bounces, and when a recipient marks your email as spam. For more information, see Using Amazon SES configuration sets (p. 249).

- When you send email through Amazon SES, it's important to monitor the bounces and complaints for your account. Amazon SES includes a reputation dashboard that you can use to keep track of the bounces and complaints for your account. For more information, see Using the reputation dashboard to track bounce and complaint rates (p. 369). You can also create CloudWatch alarms that alert you
Setting up email with Amazon SES

To set up email with Amazon Simple Email Service (Amazon SES), you need to perform the following tasks:

- Before you can access Amazon SES or other AWS services, you need to set up an AWS account. For more information, see Signing up for AWS (p. 46).
- Before you send email through Amazon SES, you need to verify that you own the "From" address. If your account is still in the Amazon SES sandbox, you also need to verify your "To" addresses. You can verify email addresses or entire domains. For more information, see Verifying identities in Amazon SES (p. 47).

The following tasks are optional depending on what you want to do:

- If you want to access Amazon SES through the Amazon SES API, whether by the Query (HTTPS) interface or indirectly through an AWS SDK, the AWS Command Line Interface or the AWS Tools for Windows PowerShell, you need to obtain your AWS access keys. For more information, see Getting your AWS access keys (p. 65).
- If you want to call the Amazon SES API without handling the low-level details of the Query interface, you can use an AWS SDK. For more information, see Downloading an AWS SDK (p. 65).
- If you want to access Amazon SES through its SMTP interface, you need to obtain your SMTP user name and password. Your SMTP credentials are different from your AWS credentials. For more information, see Getting your SMTP credentials for Amazon SES (p. 71).
- When you first sign up for Amazon SES, your account is in the Amazon SES sandbox. In the sandbox, you can send emails using the same email-sending methods as any other Amazon SES user, except that you can only send 200 emails per 24-hour period at a maximum rate of one email per second, and you can only send emails to addresses you have verified. To increase your sending quotas and to send email to unverified email addresses, see Moving out of the Amazon SES sandbox (p. 71).
- If you want your emails to pass Domain-based Message Authentication, Reporting and Conformance (DMARC) authentication based on Sender Policy Framework (SPF), configure your identity to send from a custom MAIL FROM domain as described in Setting up a custom MAIL FROM domain (p. 65).

Signing up for AWS

You have to create an AWS account before you can use Amazon SES or other AWS services.

To create an AWS account

1. In a web browser, go to https://aws.amazon.com/ses.
2. Choose Create an AWS Account.
3. Follow the on-screen instructions.

Next steps

After you create your AWS account, you can start setting up Amazon SES.
Verifying identities

To start sending email with Amazon SES, you first have to verify an identity (p. 47). An identity is an email address or domain that your email is sent from.

To interact with Amazon SES, you need to obtain the IAM credentials (p. 65) for your account.

When you first start using Amazon SES, your account is in the Amazon SES sandbox. In the sandbox, you have full access to the Amazon SES API and SMTP interface. However, the following restrictions are in effect:

- You can only send email to the Amazon SES mailbox simulator (p. 181), and to verified email identities (p. 47).
- You can send a maximum of 200 messages per 24-hour period.
- You can send a maximum of one message per second.

For information about moving out of the sandbox, see Moving out of the Amazon SES sandbox (p. 71).

Verifying identities in Amazon SES

In Amazon SES, an identity is an email address or domain that you use to send email. Before you can send an email using Amazon SES, you must verify each identity that you're going to use as a "From", "Source", "Sender", or "Return-Path" address to prove that you own it. If your account is still in the Amazon SES sandbox, you also need to verify any email addresses that you send emails to, except for email addresses provided by the Amazon SES mailbox simulator (p. 181).

You can verify an identity by using the Amazon SES console or the Amazon SES API.

Verifying email addresses in Amazon SES

Amazon SES requires that you verify your identities (the domains or email addresses that you send email from) to confirm that you own them, and to prevent unauthorized use. This section includes information about verifying email address identities. For information about verifying domain identities, see the section called “Verifying domains” (p. 59).

Consider the following factors when you verify email addresses for use with Amazon SES:

- You must verify each identity that you use as a "From," "Source," "Sender," or "Return-Path" address. You can, however, add a label to an email address that has already been verified without performing any additional verification steps (see the information later in this list).
- Email addresses are case sensitive. If you verify sender@example.com, you can't send email from sender@example.com unless you verify sender@example.com as well.
- If you verify both an email address and the domain that address belongs to, the settings for the email address override those of the domain. For example, if DomainKeys Identified Mail (DKIM) is enabled for the domain example.com, but not for sender@example.com, emails sent from sender@example.com aren't DKIM signed.
- Amazon SES has endpoints in multiple AWS Regions, and the verification status of the email address is separate for each region. If you want to send email from the same identity in more than one region, you must verify that identity in each region. For information about using Amazon SES in multiple regions, see Regions (p. 457).
- In each AWS Region, you can verify up to 10,000 identities (email addresses or domains, in any combination).
- You can add labels to verified email addresses without performing additional verification steps. To add a label to an email address, add a plus sign (+) between the account name and the "at" sign (@), followed by a text label. For example, if you already verified sender@example.com, you can use sender+myLabel@example.com as the "From" or "Return-Path" address for your emails. You can use this
feature to implement Variable Envelope Return Path (VERP). Then you can use VERP to detect and remove undeliverable email addresses from your mailing lists.

- You can customize the messages that are sent to the email addresses you attempt to verify. For more information, see Using custom verification email templates (p. 52).

### Verifying an email address

You can verify email addresses by using the Amazon SES console, the VerifyEmailIdentity Amazon SES API v1 operation, or the CreateEmailIdentity Amazon SES API v2 operation. For enhanced capabilities, we suggest using Amazon SES API v2.

#### Verifying an email address using the Amazon SES console

Complete the procedure in this section to verify an email address using the Amazon SES console.

**To verify an email address using the Amazon SES console**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the console, use the region selector to choose the AWS Region where you want to verify the email address, as shown in the following image.

![Amazon SES console](image)

   **Note**
   To verify an email address for use in more than one region, repeat the procedure in this section for each region.
3. In the navigation pane, under **Identity Management**, choose **Email Addresses**.
4. Choose **Verify a New Email Address**.
5. In the **Verify a New Email Address** dialog box, type your email address in the **Email Address** field, and then choose **Verify This Email Address**.
6. Check the inbox for the email address that you're verifying. You'll receive a message with the following subject line: "Amazon Web Services - Email Address Verification Request in region RegionName," where RegionName is the name of the AWS Region you selected in step 2.

   Click the link in the message.

   **Note**
   The link in the verification message expires 24 hours after the message was sent. If 24 hours have passed since you received the verification email, repeat steps 1–5 to receive a verification email with a valid link.

7. In the Amazon SES console, under **Identity Management**, choose **Email Addresses**. In the list of email addresses, locate the email address you're verifying. If the email address was verified, the value in the **Status** column is "verified".
Verify an email address using the Amazon SES API

You can use the VerifyEmailIdentity Amazon SES API v1 operation or the CreateEmailIdentity Amazon SES API v2 operation to create a new email identity. An easy way to call this operation for individual identities is to use the AWS CLI.

To set a default configuration set while verifying an email identity, see Managing Amazon SES default configuration sets (p. 252).

Note
Before you complete the procedure in this section, you have to install and configure the AWS CLI. For more information, see the AWS Command Line Interface User Guide.

To verify an email address by using the AWS CLI

1. At the command line, enter the following command to use the VerifyEmailIdentity Amazon SES API v1 operation:

   ```
   aws ses verify-email-identity --email-address sender@example.com
   ```

   In the preceding command, replace `sender@example.com` with the email address that you want to verify.

   At the command line, enter the following command to use the CreateEmailIdentity Amazon SES API v2 operation:

   ```
   aws sesv2 create-email-identity --email-identity sender@example.com
   ```

   In the preceding command, replace `sender@example.com` with the email address that you want to verify.

   If the command executes successfully, it exits without providing any output.

2. Check the inbox for the email address that you’re verifying. You’ll receive a message with the following subject line: "Amazon Web Services - Email Address Verification Request in region RegionName," where RegionName is the name of the AWS Region that you attempted to verify the email address in.

   Open the message, and then click the link in it.

   Note
   The link in the verification message expires 24 hours after the message was sent. If 24 hours have passed since you received the verification email, repeat steps 1–5 to receive a verification email with a valid link.

3. In the Amazon SES console, under Identity Management, choose Email Addresses. In the list of email addresses, locate the email address you’re verifying. If the email address was verified, the value in the Status column is "verified".

For a script that can be used to verify several email identities in a single operation, see the section called “Verify multiple email addresses” (p. 450).

Troubleshoot email address verification

If you attempted to verify an email address, but didn’t receive a verification email from AWS, try the following troubleshooting steps:

- Check the Junk Mail folder in your email client.
- Ensure that your email client isn’t applying rules that automatically move certain messages to a folder other than your inbox.
• In your email client, add no-reply-aws@amazon.com to your address book or Safe Senders list. You can also ask your system administrator to whitelist incoming email from no-reply-aws@amazon.com.

• With an email address that uses a different email service provider (such as a personal email address), send a message to the address you want to verify. Ensure that the address you want to verify receives the message. This step is especially important if you recently set up your own domain. Occasionally, new domains might not be correctly configured to receive incoming email.

Alternatively, try to verify an email address that you know is able to receive email, such as a personal email address. If you receive the verification email at your personal address, it might indicate that there is an issue on the other domain.

If these tests show email isn't being received at the address you attempted to verify, consult your system administrator or email service provider for further assistance.

Listing email identities in Amazon SES

You can display a list of email identities by using the Amazon SES console or the ListIdentities API operation.

Viewing a list of email identities in Amazon SES

You can use the Amazon SES console and API to view a list of email addresses that are verified or are pending verification, as well as those that failed the verification process.

To view a list of verified email addresses

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.

2. In the console, use the region selector to choose the AWS Region where you want to list email identities, as shown in the following image.

   ![Amazon SES Console](image)

   **Note**
   These procedures only display a list of email addresses for the selected AWS Region.

3. In the navigation pane, under Identity Management, choose Email Addresses.

   The Email Addresses page displays a list of email addresses that are verified, that are pending verification, and that failed the verification process. Click an email address to view additional information about it.

Viewing a list of email identities using the Amazon SES API

Use the ListIdentities API operation to view a list of all email identities, regardless of their statuses. You can also use the GetIdentityVerificationAttributes operation to find the verification status of a given identity.
To view a list of identities by using the AWS CLI, type the following command at the command line: `aws ses list-identities`

When you execute the `ListIdentities` operation, it returns a list of all of the identities in your Amazon SES account, regardless of their verification statuses. To see the verification status for one or more identities, use the `GetIdentityVerificationAttributes` operation. To find the verification status of an identity using the AWS CLI, type the following command at the command line: `aws ses get-identity-verification-attributes --identities "sender@example.com"

Replace `sender@example.com` in the preceding command with the identity that you want to find the verification status of. You can also use this command to find the verification statuses of multiple identities in a single API call. For example, to find the status of the domain `example.com` and the email address `sender@example.co.uk`, type the following command: `aws ses get-identity-verification-attributes --identities "example.com" "sender@example.co.uk"

### Deleting an email identity in Amazon SES

If you no longer need to use a verified email address, you can delete it by using the Amazon SES console or the `DeleteIdentity` API operation.

**Warning**

This action can't be undone. However, you can repeat the verification process for an identity that was previously deleted.

### Deleting an email identity in Amazon SES

To remove verified email addresses

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the console, use the region selector to choose the AWS Region where you want to delete an email identity, as shown in the following image.

![Amazon SES console](image)

**Note**

These procedures only delete the email address in the selected AWS Region. To delete an email address that was verified in more than one region, repeat the procedure in this section for each region.

3. Select each email address that you want to remove, and then choose Remove.

### Deleting an email identity using the Amazon SES API

Use the `DeleteIdentity` API operation to delete email address and domain identities.

To delete an identity using the AWS CLI, type the following command at the command line: `aws ses delete-identity --identity "sender@example.com"`
Replace `sender@example.com` in the preceding command with the identity that you want to delete.

**Using custom verification email templates**

When you attempt to verify an email address, Amazon SES sends an email to that address that resembles the example shown in the following image.

```
Dear Amazon Web Services Customer,

We have received a request to authorize this email address for use with Amazon SES and Amazon Pinpoint in region US West (Oregon). If you requested this verification, please go to the following URL to confirm that you are authorized to use this email address:

https://emailverification.us-west-2.amazonaws.com/0AWSAccessKeyId-bAUKQKUYEXAMPLE&Context=10987654312&
Identity.IdentityName=recipient@example.com&Identity.IdentityType=EmailAddress&Namespace=Amazon&Operation=ConfirmVerification&
Signature=TJD4nYy17fC5qg9b0Qy2gqZbhW1uYEXAMPLEĿSignatureMethod= HMACSHA256&SignatureVersion=2&
Timestamp=2017-12-06T19%3A53%3A12Z

Your request will not be processed unless you confirm the address using this URL. This link expires 24 hours after your original verification request.

If you did not request to verify this email address, do not click on the link. Please note that, many times, the situation isn’t a phishing attempt, but rather a misunderstanding of how to use our service, or someone setting up email sending capabilities on your behalf as part of a legitimate service, but without having fully communicated the procedure first. If you are still concerned, please forward this notification to aws-email-domain-verification@amazon.com and let us know in the forward that you did not request the verification.


Sincerely,
The Amazon Web Services Team.
```

Several Amazon SES customers build applications (such as email marketing suites or ticketing systems) that send email through Amazon SES on behalf of their own customers. For the end users of these applications, the email verification process can be confusing: the verification email uses Amazon SES branding, rather than the branding of the application, and those end users never signed up to use Amazon SES directly.

If your Amazon SES use case requires your customers to have their email addresses verified for use with Amazon SES, you can create customized verification emails. These customized emails help reduce customer confusion and increase the rates at which your customers complete the registration process.

**Note**

To use this feature, your Amazon SES account has to be out of the sandbox. For more information, see Moving out of the Amazon SES sandbox (p. 71).

**Topics in this section:**

- Creating a custom verification email template (p. 52)
- Editing a custom verification email template (p. 54)
- Sending verification emails using custom templates (p. 54)
- Custom verification email frequently asked questions (p. 55)

**Creating a custom verification email template**

To create a custom verification email, use the `CreateCustomVerificationEmailTemplate` API operation. This operation takes the following inputs:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TemplateName</td>
<td>The name of the template. The name you specify must be unique.</td>
</tr>
<tr>
<td>FromEmailAddress</td>
<td>The email address that the verification email is sent from. The address or domain you specify must be verified for use with your Amazon SES account. <strong>Note</strong> The <code>FromEmailAddress</code> attribute doesn’t support display names (also known as “friendly from” names).</td>
</tr>
<tr>
<td>Attribute</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>TemplateSubject</td>
<td>The subject line of the verification email.</td>
</tr>
<tr>
<td>TemplateContent</td>
<td>The body of the email. The email body can contain HTML, with certain</td>
</tr>
<tr>
<td></td>
<td>restrictions. For more information, see Custom verification email</td>
</tr>
<tr>
<td></td>
<td>frequently asked questions (p. 55).</td>
</tr>
<tr>
<td>SuccessRedirectionURL</td>
<td>The URL that users are sent to if their email addresses are successfully</td>
</tr>
<tr>
<td></td>
<td>verified.</td>
</tr>
<tr>
<td>FailureRedirectionURL</td>
<td>The URL that users are sent to if their email addresses are not successfully</td>
</tr>
<tr>
<td></td>
<td>verified.</td>
</tr>
</tbody>
</table>

You can use the AWS SDKs or the AWS CLI to create a custom verification email template with the CreateCustomVerificationEmailTemplate operation. To learn more about the AWS SDKs, see Tools for Amazon Web Services. For more information about the AWS CLI, see AWS Command Line Interface.

The following section includes procedures for creating a custom verification email using the AWS CLI. These procedures assume that you have installed and configured the AWS CLI. For more information about installing and configuring the AWS CLI, see the AWS Command Line Interface User Guide.

**Note**
To complete the procedure in this section, you must use version 1.14.6 or later of the AWS CLI. For best results, upgrade to the latest version of the AWS CLI. For more information about updating the AWS CLI, see Installing the AWS Command Line Interface in the AWS Command Line Interface User Guide.

1. In a text editor, create a new file. Paste the following content into the editor:

```json
{
  "TemplateName": "SampleTemplate",
  "FromEmailAddress": "sender@example.com",
  "TemplateSubject": "Please confirm your email address",
  "TemplateContent": "<html>
  <head></head>
  <body style='font-family:sans-serif; '>
    <h1 style='text-align:center'>Ready to start sending email with ProductName?</h1>
    <p>We here at Example Corp are happy to have you on board! There's just one last step to complete before you can start sending email. Just click the following link to verify your email address. Once we confirm that you're really you, we'll give you some additional information to help you get started with ProductName.</p>
  </body>
</html>

"SuccessRedirectionURL": "https://www.example.com/verifysuccess",
"FailureRedirectionURL": "https://www.example.com/verifyfailure"
}
```

**Important**
To make the preceding example easier to read, the TemplateContent attribute contains line breaks. If you paste the preceding example into your text file, remove the line breaks before proceeding.

Replace the values of TemplateName, FromEmailAddress, TemplateSubject, TemplateContent, SuccessRedirectionURL, and FailureRedirectionURL with your own values.
Verifying identities

Note

The email address that you specify for the FromEmailAddress parameter has to be verified, or has to be an address on a verified domain. For more information, see Verifying identities in Amazon SES (p. 47).

When you finish, save the file as customverificationemail.json.

2. At the command line, type the following command to create the custom verification email template:

```
aws ses create-custom-verification-email-template --cli-input-json file://customverificationemail.json
```

3. Optionally, you can confirm that the template was created by typing the following command:

```
aws ses list-custom-verification-email-templates
```

Editing a custom verification email template

You can edit a custom verification email template using the UpdateCustomVerificationEmailTemplate operation. This operation accepts the same inputs as the CreateCustomVerificationEmailTemplate operation (that is, the TemplateName, FromEmailAddress, TemplateSubject, TemplateContent, SuccessRedirectionURL, and FailureRedirectionURL attributes). However, with the UpdateCustomVerificationEmailTemplate operation, none of these attributes are required. When you pass a value for TemplateName that is the same as the name of an existing custom verification email template, the attributes you specify overwrite the attributes that were originally in the template.

Sending verification emails using custom templates

After you create at least one custom verification email template, you can send it to your customers by calling the SendCustomVerificationEmail API operation. You can call the SendCustomVerificationEmail operation by using any of the AWS SDKs or the AWS CLI. The SendCustomVerificationEmail operation takes the following inputs:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EmailAddress</td>
<td>The email address that is being verified.</td>
</tr>
<tr>
<td>TemplateName</td>
<td>The name of the custom verification email template that is sent to email address that is being verified.</td>
</tr>
<tr>
<td>ConfigurationSetName</td>
<td>(Optional) The name of a configuration set to use when sending the verification email.</td>
</tr>
</tbody>
</table>

For example, assume your customers register for your service using a form in your application. When the customer completes the form and submits it, your application calls the SendCustomVerificationEmail operation, passing the customer's email address and the name of the template you want to use.

Your customer receives an email that uses the customized email template you created. Amazon SES automatically adds a unique link to the recipient, as well as a brief disclaimer. The following image shows a sample verification email that uses the template created in Creating a custom verification email template (p. 52).
Custom verification email frequently asked questions

This section contains answers to frequently asked questions about the custom verification email template feature.

Q1. How many custom verification email templates can I create?

You can create up to 50 custom verification email templates per Amazon SES account.

Q2. How do custom verification emails appear to recipients?

Custom verification emails include the content you specified when you created the template, followed by a link that recipients must click to verify their email addresses.

Q3. Can I preview the custom verification email?

To preview a custom verification email, use the `SendCustomVerificationEmail` operation to send a verification email to an address you own. If you do not click the verification link, Amazon SES does not create a new identity. If you do click the verification link, you can optionally delete the newly created identity using the `DeleteIdentity` operation.

Q4. Can I include images in my custom verification email templates?

You can embed images in the HTML for your templates by using Base64 encoding. When you embed images in this way, Amazon SES automatically converts them into attachments. You can encode an image at the command line by issuing one of the following commands:

Linux, macOS, or Unix

```
base64 -i imagefile.png | tr -d '
' > output.txt
```

Windows

```
certutil -encodehex -f imagefile.png output.txt 0x40000001
```

Replace `imagefile.png` with the name of the file you want to encode. In both of the commands above, the Base64 encoded image is saved to `output.txt`.

You can embed the Base64-encoded image by including the following in the HTML for the template:

```
<img src="data:image/png;base64,base64EncodedImage" />
```

In the example above, replace `png` with the file type of the encoded image (such as `jpg` or `gif`), and replace `base64EncodedImage` with the Base64 encoded image (that is, the contents of `output.txt` from one of the preceding commands).

Q5. Are there any limits to the content that I can include in custom verification email templates?

Custom verification email templates can't exceed 10 MB in size. Additionally, custom verification email templates that contain HTML can only use the tags and attributes listed in the following table:
<table>
<thead>
<tr>
<th>HTML tag</th>
<th>Allowed attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>abbr</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>acronym</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>address</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>area</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>b</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>bdo</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>big</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>blockquote</td>
<td>cite, class, id, style, title</td>
</tr>
<tr>
<td>body</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>br</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>button</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>caption</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>center</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>cite</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>code</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>col</td>
<td>class, id, span, style, title, width</td>
</tr>
<tr>
<td>colgroup</td>
<td>class, id, span, style, title, width</td>
</tr>
<tr>
<td>dd</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>del</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>dfn</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>dir</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>div</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>dl</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>dt</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>em</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>fieldset</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>font</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>form</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>h1</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>h2</td>
<td>class, id, style, title</td>
</tr>
<tr>
<td>h3</td>
<td>class, id, style, title</td>
</tr>
</tbody>
</table>
## HTML tag | Allowed attributes
--- | ---
h4 | class, id, style, title
h5 | class, id, style, title
h6 | class, id, style, title
head | class, id, style, title
hr | class, id, style, title
html | class, id, style, title
i | class, id, style, title
img | align, alt, class, height, id, src, style, title, width
input | class, id, style, title
ins | class, id, style, title
kbd | class, id, style, title
label | class, id, style, title
legend | class, id, style, title
li | class, id, style, title
map | class, id, style, title
menu | class, id, style, title
ol | class, id, start, style, title, type
optgroup | class, id, style, title
option | class, id, style, title
p | class, id, style, title
pre | class, id, style, title
q | cite, class, id, style, title
s | class, id, style, title
samp | class, id, style, title
select | class, id, style, title
small | class, id, style, title
span | class, id, style, title
strike | class, id, style, title
strong | class, id, style, title
sub | class, id, style, title
### HTML tag | Allowed attributes
--- | ---
sup | class, id, style, title
table | class, id, style, summary, title, width
tbody | class, id, style, title
td | abbr, axis, class, colspan, id, rowspan, style, title, width
textarea | class, id, style, title
tfoot | class, id, style, title
th | abbr, axis, class, colspan, id, rowspan, scope, style, title, width
thead | class, id, style, title
tr | class, id, style, title
tt | class, id, style, title
u | class, id, style, title
ul | class, id, style, title, type
var | class, id, style, title

**Note**

Custom verification email templates can't include comment tags.

**Q6. How many verified email addresses can exist in my account?**

Your Amazon SES account can have up to 10,000 verified identities in each AWS Region. In Amazon SES, identities include both verified domains and email addresses.

**Q7. Can I create custom verification email templates using the Amazon SES console?**

Currently, it is only possible to create, edit, and delete custom verification emails using the Amazon SES API.

**Q8. Can I track open and click events that occur when customers receive custom verification emails?**

Custom verification emails cannot include open or click tracking.

**Q9. Can custom verification emails include custom headers?**

Custom verification emails cannot include custom headers.

**Q10. Can I remove the text that appears at the bottom of custom verification emails?**

The following text is automatically added to the end of every custom verification email and cannot be removed:

*If you did not request to verify this email address, please disregard this message. If you have any concerns, please forward this message to the following email address along with your questions or comments.*
The email address link in this text refers to aws-email-domain-verification@amazon.com, an inbox that is actively monitored by the Amazon SES team.

Q11. Are custom verification emails DKIM-signed?

In order for verification emails to be DKIM-signed, the email address that you specify in the FromEmailAddress attribute when you create the verification email template must be configured to generate a DKIM signature. For more information about setting up DKIM for domains and email addresses, see the section called “Authenticating Email with DKIM” (p. 130).

Q12. Why don’t the custom verification email template API operations appear in the SDK or CLI?

If you’re unable to use the custom verification email template operations in an SDK or the AWS CLI, you may be using an older version of the SDK or CLI. The custom verification email template operations are available in the following SDKs and CLIs:

- Version 1.14.6 or later of the AWS Command Line Interface
- Version 3.3.205.0 or later of the AWS SDK for .NET
- Version 1.3.20170531.19 or later of the AWS SDK for C++
- Version 1.12.43 or later of the AWS SDK for Go
- Version 1.11.245 or later of the AWS SDK for Java
- Version 2.166.0 or later of the AWS SDK for JavaScript
- Version 3.45.2 or later of the AWS SDK for PHP
- Version 1.5.1 or later of the AWS SDK for Python (Boto)
- Version 1.5.0 or later of the aws-sdk-ses gem in the AWS SDK for Ruby

Q13. Why do I receive ProductionAccessNotGranted errors when I send custom verification emails?

The ProductionAccessNotGranted error indicates that your account is still in the Amazon SES sandbox. You can only send custom verification emails if your account has been removed from the sandbox. For more information, see Moving out of the Amazon SES sandbox (p. 71).

Verifying domains in Amazon SES

Amazon SES requires that you verify your email address or domain, to confirm that you own it and to prevent others from using it. When you verify an entire domain, you are verifying all email addresses from that domain, so you don’t need to verify email addresses from that domain individually. For example, if you verify the domain example.com, you can send email from user1@example.com, user2@example.com, or any other user at example.com.

You can manage your verified domains by using the Amazon SES console or the Amazon SES API. For a complete description of API actions related to domain verification, go to the Amazon Simple Email Service API Reference. This section, which demonstrates the actions using the Amazon SES console, contains the following topics:

- Verifying a domain with Amazon SES (p. 60)
- Listing domain identities in Amazon SES (p. 62)
- Deleting a domain identity in Amazon SES (p. 62)
- Amazon SES domain verification revocation (p. 63)
- Amazon SES domain verification TXT records (p. 63)

Important notes about domain verification are as follows:
• Amazon SES has endpoints in multiple AWS regions, and domain verification applies to each AWS region separately. You must perform the entire domain verification procedure for each region in which you want to send from a given domain. If you want to verify the same domain in multiple regions and your DNS provider does not allow you to have multiple TXT records with the same name, see the workarounds in Common domain verification problems (p. 471).

• If you verify a domain with Amazon SES, you can send from any subdomain of that domain without specifically verifying the subdomain. For example, if you verify example.com, you do not need to verify a.example.com or a.b.example.com. As specified in RFC 1034, each DNS label can have up to 63 characters and the whole domain name must not exceed a total length of 255 characters.

• If you verify a domain, subdomain(s), and/or email address(es) that share a root domain, the verified identity settings (such as feedback notifications and Easy DKIM) apply at the most granular level you verified. That is:
  • Verified email address settings override verified domain settings.
  • Verified subdomain settings override verified domain settings, with lower-level subdomain settings overriding higher-level subdomain settings.

For example, assume you verify user@a.b.example.com, a.b.example.com, b.example.com, and example.com. These are the verified identity settings that will be used in the following scenarios:
  • Emails sent from user@example.com (an address that is not specifically verified) will use the settings for example.com.
  • Emails sent from user@a.b.example.com (an address that is specifically verified) will use the settings for user@a.b.example.com.
  • Emails sent from user@b.example.com (an address that is not specifically verified) will use the settings for b.example.com.
  • Domain names are case-insensitive. If you verify example.com, you can send from EXAMPLE.com also.
  • In each AWS Region, you can verify as many as 10,000 identities (domains and email addresses, in any combination).

Verifying a domain with Amazon SES

The following procedure shows you how to verify a domain using the Amazon SES console. If you want to use the Amazon SES API v1 instead, see the Amazon Simple Email Service API Reference.

To set a default configuration set while verifying an email identity, see Managing Amazon SES default configuration sets (p. 252).

Note
As an alternative to completing the procedure in this section, you can also enable Easy DKIM (p. 131). When Amazon SES detects that you’ve added the DKIM records to the DNS configuration for a domain, you can start sending email from that domain, even if you haven’t already completed the procedure in this section.

To verify a domain

1. Go to your verified domain list in the Amazon SES console, or follow these instructions to navigate to it:
   a. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
   b. In the navigation pane, under Identity Management, choose Domains.
2. Choose Verify a New Domain.
3. In the Verify a New Domain dialog box, enter the domain name.
Tip
If your domain is www.example.com, enter example.com as your domain. The "www." part isn't necessary, and the domain verification process won't succeed if you include it.

4. If you want to set up DKIM signing for this domain, choose Generate DKIM Settings. For information about DKIM signing, see Authenticating Email with DKIM in Amazon SES (p. 130).

5. Choose Verify This Domain.

6. In the Verify a New Domain dialog box, you will see a Domain Verification Record Set containing a Name, a Type, and a Value. (This information will also be available by choosing the domain name after you close the dialog box.)

To complete domain verification, add a TXT record with the displayed Name and Value to your domain's DNS server. For information about Amazon SES TXT records and general guidance about how to add a TXT record to a DNS server, see Amazon SES domain verification TXT records (p. 63).

In particular:

- If your DNS provider does not allow underscores in record names, you can omit _amazonses from the Name.
- To help you easily identify this record within your domain's DNS settings, you can optionally prefix the Value with amazonses:
- Some DNS providers automatically append the domain name to DNS record names. To avoid duplication of the domain name, you can add a period to the end of the domain name in the DNS record. This indicates that the record name is fully qualified and the DNS provider need not append an additional domain name.

7. If Route 53 provides the DNS service for the domain that you're verifying, and you're signed in to the AWS Management Console under the same account that you use for Route 53, then Amazon SES gives you the option of updating your DNS server immediately from within the Amazon SES console.

If you use a different DNS provider, the procedures for updating the DNS records vary depending on which DNS or web hosting provider you use. The following table lists links to the documentation for several common providers. This list isn't exhaustive and inclusion in this list isn't an endorsement or recommendation of any company's products or services. If your provider isn't listed in the table, you can probably use the domain with Amazon SES.
When verification is complete, the domain's status in the Amazon SES console changes from "pending verification" to "verified," and you receive a notification email from Amazon SES.

8. You can now use Amazon SES to send email from any address in the verified domain. To send a test email, check the box next to the verified domain, and then choose **Send a Test Email**.

If the DNS settings are not correctly updated, you will receive a domain verification failure email from Amazon SES, and the domain will display a status of **failed** on the **Domains** tab. If this happens, complete the steps on the troubleshooting page at Common domain verification problems (p. 471). After you verify that your TXT was created correctly, choose the **retry** link next to the **failed** status notification to restart the domain verification process.

### Listing domain identities in Amazon SES

To view your verified domains, follow the procedure below.

**To view your verified domains**

1. Go to your verified domain list in the Amazon SES console, or follow these instructions to navigate to it:
   a. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
   b. In the navigation pane, under **Identity Management**, choose **Domains**.
2. In the list of verified domains, you can expand one or more domains to view the details.

### Deleting a domain identity in Amazon SES

To remove a verified domain, follow the procedure below.

**To remove a verified domain**

1. Go to your verified domain list in the Amazon SES console, or follow these instructions to navigate to it:
   a. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.

---

### DNS/Hosting Provider | Documentation Link
---
| GoDaddy | Add a TXT record |
| Dreamhost | How do I add custom DNS records? |
| Cloudflare | Managing DNS records in CloudFlare |
| HostGator | Manage DNS Records with HostGator/eNom |
| Namecheap | How do I add TXT/SPF/DKIM/DMARC records for my domain? |
| Names.co.uk | Changing your domain’s DNS settings |
| Wix | Adding or Updating TXT Records in Your Wix Account |
b. In the navigation pane, under Identity Management, choose Domains.

2. Check the box beside each domain that you want to remove, and then choose Remove.

3. You will no longer be able to send email from the removed domain.

Amazon SES domain verification revocation

Amazon SES periodically reviews domain verification status, and revokes verification in cases where it is no longer valid. If Amazon SES is unable to detect the TXT record information required to confirm ownership of a domain, you will receive an Amazon SES Domain Verification REVOCATION WARNING email from Amazon SES.

If you restore the TXT record information to your domain's DNS server within 72 hours, you will receive an Amazon SES Domain Verification REVOCATION CANCELLATION email from Amazon SES.

Note
You can review the required TXT record information in the Amazon SES console by using the following instructions. In the navigation pane, under Identity Management, choose Domains. In the list of domains, choose (not just expand) the domain to display the domain verification settings, which include the TXT record name and value.

If you do not restore the TXT record information to your domain's DNS server within 72 hours, you will receive an Amazon SES Domain Verification REVOCATION email from Amazon SES, the domain will be removed from the list of Verified Senders on the Domains tab, and you will no longer be able to send from the domain.

To reverify a domain for which verification has been revoked, you must restart the verification procedure from the beginning, just as if the revoked domain were an entirely new domain.

Amazon SES domain verification TXT records

Your domain is associated with a set of Domain Name System (DNS) records that you manage through your DNS provider. A TXT record is a type of DNS record that provides additional information about your domain. Each TXT record consists of a name and a value.

When you initiate domain verification using the Amazon SES console or API, Amazon SES gives you the name and value to use for the TXT record. For example, if your domain is example.com, the TXT record settings that Amazon SES generates will look similar to the following example:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>_amazonses.example.com</td>
<td>TXT</td>
<td>pmBGN/7MjnfhTKUZ06Enqq1PeGUaOkw8lGhcfwefcHU=</td>
</tr>
</tbody>
</table>

Add a TXT record to your domain's DNS server using the specified Name and Value. Amazon SES domain verification is complete when Amazon SES detects the existence of the TXT record in your domain's DNS settings.

Note
Some DNS providers automatically append your domain name to DNS record names. To avoid duplication of the domain name, you can add a period (.) to the end of the domain name in the DNS record, or omit your domain from the record name. For more information, see the documentation for your DNS provider.

If your DNS provider does not allow DNS record names to contain underscores, you can omit _amazonses from the Name. In that case, for the preceding example, the TXT record name would be example.com.
instead of _amazonses.example.com. To make the record easier to recognize and maintain, you can also optionally prefix the Value with amazonses:. In the previous example, the value of the TXT record would therefore be amazonses:pmBGN/7MjnfhTKUZ06Enqq1PeGUaOkw8lGhcfwefcHU=.

Note
Amazon SES previously allowed TXT record names to contain amazonses without an underscore. If you have already verified a domain and your TXT record contains amazonses without an underscore, your domain will continue to be verified; there is no action required on your part. However, any new domains that you verify will require that amazonses in the TXT record name is either preceded by an underscore, or _amazonses is removed from the TXT record name entirely.

You can find troubleshooting information and instructions on how to check your domain verification settings in Common domain verification problems (p. 471).

Adding a TXT Record to Your Domain's DNS Server

The procedure for adding TXT records to your domain's DNS server depends on who provides your DNS service. Your DNS provider might be Amazon Route 53 or another domain name registrar such as GoDaddy. This section provides procedures for adding a TXT record to Route 53, as well as generic procedures that apply to other DNS providers.

Procedures for Amazon Route 53

When you begin the process of verifying a new domain (p. 60) for use with Amazon SES, you can automatically add the domain verification TXT record to your Route 53 configuration. However, if you choose not to add the TXT record automatically, you can add the TXT record to your Route 53 configuration manually by completing the procedure in this section.

To add a TXT record to the DNS record for your Route 53-managed domain

1. Open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. Under Identity Management, choose Domains.
3. Choose the domain that you want to verify.
4. Expand the Verification section. Copy the value shown next to TXT Value.
6. In the navigation pane, choose Hosted zones.
7. Choose the domain that you want to add a TXT record to.
8. Choose Create record.
9. For Routing policy, choose Simple routing, and then choose Next.
10. Choose Define simple record.
11. In the Define simple record pane, do the following:
   a. For Record name, enter _amazonses.
   b. For Value/Route traffic to, choose IP address or another value depending on the record type. Then, in the text area, paste the TXT record value that you copied from the Amazon SES console.
   c. For Record type, choose TXT.
   d. For TTL (seconds), type 1800.
   e. Choose Define simple record.
12. Wait for about 30 minutes for the record to propagate. Then, on the Domains page in the Amazon SES console, check the value in the Status column next to the domain. If the status is "pending verification," wait for another 30 minutes, and then choose refresh (⟲). Repeat this process until the value in the status column is "verified."
Generic procedures for other DNS providers

The procedures for adding TXT records to the DNS configurations vary from provider to provider. For specific steps, consult your DNS provider's documentation. The procedure in this section gives a basic overview of the steps you take when adding a TXT record to the DNS configuration for your domain.

To add a TXT record to your domain's DNS server (general procedure)

1. Go to your DNS provider’s management console and sign in to your account.
2. Find the page for updating your domain's DNS records. This page might have a name similar to one of the following examples: DNS Records, DNS Zone File, or Advanced DNS. See the documentation provided by your DNS provider for more information.
3. Add a TXT record with the name and value provided by Amazon SES.
   
   **Important**
   
   Some DNS providers, such as GoDaddy, automatically append the domain name to the end of DNS records. Adding a record that already contains the domain name (such as `_amazonses.example.com`) might result in the duplication of the domain name (such as `_amazonses.example.com.example.com`). To avoid duplication of the domain name, add a period to the end of the domain name in the DNS record, or just omit your domain from the record name. See the documentation provided by your DNS provider for more information.
4. Save your changes. DNS record updates can take up to 48 hours to take effect, but they often take effect much sooner. You can verify that the TXT record is correctly published by using the procedure in Checking domain verification settings (p. 472).

Getting your AWS access keys

After you've signed up for Amazon SES, you'll need to obtain your AWS access keys if you want to access Amazon SES through the Amazon SES API, whether by the Query (HTTPS) interface directly or indirectly through an AWS SDK, the AWS Command Line Interface, or the AWS Tools for Windows PowerShell. AWS access keys consist of an access key ID and a secret access key.

For more information about the types of security keys that you can use in Amazon SES, see Types of Amazon SES credentials (p. 16). For information about obtaining AWS access keys, see AWS security credentials in the AWS General Reference.

Downloading an AWS SDK

If you want to call the Amazon SES API without having to handle low-level details like assembling raw HTTP requests, you can use an AWS SDK. The AWS SDKs provide functions and data types that encapsulate the functionality of Amazon SES and other AWS services. To download an AWS SDK, go to SDKs.

The Getting Started section of this guide provides examples of how to send an email using the AWS SDKs for various programming languages. For more information, see Send an email through Amazon SES using an AWS SDK (p. 30).

Setting up a custom MAIL FROM domain

When an email is sent, it has two addresses that indicate its source: a From address that's displayed to the message recipient, and a MAIL FROM address that indicates where the message originated. The MAIL FROM address is sometimes called the envelope sender, envelope from, bounce address, or Return Path address. Mail servers use the MAIL FROM address to return bounce messages and other error notifications. The MAIL FROM address is usually only viewable by recipients if they view the source code for the message.
Amazon SES sets the MAIL FROM domain for the messages that you send to a default value unless you specify your own domain. This section discusses the benefits of setting up a custom MAIL FROM domain, and includes setup procedures.

Why use a custom MAIL FROM domain?

By default, messages that you send through Amazon SES use a subdomain of amazonses.com as the MAIL FROM domain. Sender Policy Framework (SPF) authentication successfully validates these messages because the default MAIL FROM domain matches the application that sent the email—in this case, Amazon SES.

While this level of authentication is sufficient for many senders, other senders prefer to set the MAIL FROM domain to a domain that they own. By setting up a custom MAIL FROM domain, your emails can comply with Domain-based Message Authentication, Reporting and Conformance (DMARC) (p. 142). DMARC enables a sender's domain to indicate that emails sent from the domain are protected by one or more authentication systems.

There are two ways to achieve DMARC validation: using Sender Policy Framework (p. 129) (SPF), and using DomainKeys Identified Mail (p. 130) (DKIM). The only way to comply with DMARC through SPF is to use a custom MAIL FROM domain, because SPF validation requires the domain in the From address to match the MAIL FROM domain. By using your own MAIL FROM domain, you have the flexibility to use SPF, DKIM, or both to achieve DMARC validation.

Choosing a MAIL FROM domain

The subdomain you use for your MAIL FROM domain has to meet the following requirements:

- The MAIL FROM domain has to be a subdomain of the verified identity (email address or domain) that you send email from. For example, mail.example.com is a valid MAIL FROM domain for the domain example.com.
- The MAIL FROM domain shouldn't be a domain that you send email from. If you have to use the MAIL FROM domain in a From address, either disable email feedback forwarding (p. 266) and receive your bounces through Amazon SNS notifications, or ensure that your MAIL FROM domain is not the destination for feedback forwarding. To determine the destination of email forwarding feedback, see Email feedback forwarding destination (p. 267).
- The MAIL FROM domain shouldn't be a domain that you use to receive email.

Configuring the MAIL FROM domain

The process of setting up a custom MAIL FROM domain requires you to add records to the DNS configuration for the domain. You have to publish an MX record so that your domain can receive the bounce and complaint notifications that email providers send you. You also have to publish an SPF record in order to prove that Amazon SES is authorized to send email from your domain.

You can set up a custom MAIL FROM domain for an entire domain, or for individual email addresses. The following procedures show how to use the Amazon SES console to configure a custom MAIL FROM domain. You can also configure a custom MAIL FROM domain using the SetIdentityMailFromDomain API operation.

Setting Up a MAIL FROM Domain for a Verified Domain

You can configure a MAIL FROM domain for an entire domain. When you do, all of the messages that you send from addresses on that domain use the same MAIL FROM domain.

To configure a verified domain to use a specified MAIL FROM domain

1. Open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, under **Identity Management**, choose **Domains**.

3. In the list of domains, confirm that the parent domain of the MAIL FROM domain is verified. If the domain isn't verified, complete the procedures at Verifying domains in Amazon SES (p. 59) to verify the domain. Otherwise, choose the domain and proceed to the next step.

4. Under **MAIL FROM Domain**, choose **Set MAIL FROM Domain**.

5. On the **Set MAIL FROM Domain** window, do the following:
   a. For **MAIL FROM domain**, enter the subdomain that you want to use as the MAIL FROM domain.
   b. For **Behavior if MX record not found**, choose one of the following options:
      - **Use region.amazonses.com as MAIL FROM** – If the custom MAIL FROM domain's MX record is not set up correctly, Amazon SES will use a subdomain of amazonses.com. The subdomain varies based on the AWS Region in which you use Amazon SES.
      - **Reject message** – If the custom MAIL FROM domain's MX record is not set up correctly, Amazon SES will return a MailFromDomainNotVerified error. Emails that you attempt to send from this domain will be automatically rejected.
   c. Choose **Set MAIL FROM Domain**. A window appears that contains the MX and SPF records that you have to add to your domain's DNS configuration. These records use the formats shown in the following table.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>subdomain.domain.com</td>
<td>MX</td>
<td>10 feedback-smtp.region.amazonses.com</td>
</tr>
<tr>
<td>subdomain.domain.com</td>
<td>TXT</td>
<td>&quot;v=spf1 include:amazonses.com ~all&quot;</td>
</tr>
</tbody>
</table>

In the preceding records, replace **subdomain.domain.com** with your MAIL FROM subdomain, and replace **region** with the name of the AWS Region where you want to verify the MAIL FROM domain (such as us-west-2, us-east-1, or eu-west-1). Note that the value of the TXT record has to include the quotation marks.

Note these values, and then proceed to the next step.

6. Publish an MX record to the DNS server of the custom MAIL FROM domain.

   **Important**
   To successfully set up a custom MAIL FROM domain with Amazon SES, you must publish exactly one MX record to the DNS server of your MAIL FROM domain. If the MAIL FROM domain has multiple MX records, the custom MAIL FROM setup with Amazon SES will fail.

   If Route 53 provides the DNS service for your MAIL FROM domain, and you are signed in to the AWS Management Console under the same account that you use for Route 53, then choose **Publish Records Using Route 53**. The DNS records are automatically applied to your domain's DNS configuration.

   If you use a different DNS provider, you have to publish the DNS records to the MAIL FROM domain's DNS server manually. The procedure for adding DNS records to your domain's DNS server varies based on your web hosting service or DNS provider.

   The procedures for publishing DNS records for your domain depend on which DNS provider you use. The following table includes links to the documentation for several common DNS providers. This list isn't a complete list of providers. If your provider isn't listed below, you can probably still set up a MAIL FROM domain. Inclusion on this list isn't an endorsement or recommendation of any company's products or services.
When Amazon SES detects that the records are in place, you receive an email informing you that your custom MAIL FROM domain was set up successfully. Depending on your DNS provider, there might be a delay of up to 72 hours before Amazon SES detects the MX record.

### Setting Up a MAIL FROM Domain for a Verified Email Address

You can also set up a custom MAIL FROM domain for a specific email address. In order to set up a custom MAIL FROM domain for an email address, you have to be able to modify the DNS records for the domain that the email address is associated with.

**Note**

You can't set up a custom MAIL FROM domain for addresses on a domain that you don't own (for example, you can't create a custom MAIL FROM domain for an address on the gmail.com domain, because you can't add the necessary DNS records to the domain).

### To configure a verified email address to use a specified MAIL FROM domain

1. Open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, under **Identity Management**, choose **Email Addresses**.
3. In the list of email addresses, confirm that the email address that you want to set up a custom MAIL FROM domain for is verified. If the email address isn't verified, complete the procedures at [Verifying email addresses in Amazon SES](p. 47) to verify the email address. Otherwise, choose the email address and proceed to the next step.
4. Under **MAIL FROM Domain**, choose **Set MAIL FROM Domain**.
5. On the **Set MAIL FROM Domain** window, do the following:
   a. For **MAIL FROM domain**, enter the subdomain that you want to use as the MAIL FROM domain.
   b. For **Behavior if MX record not found**, choose one of the following options:
• **Use region.amazonses.com as MAIL FROM** – If the custom MAIL FROM domain's MX record is not set up correctly, Amazon SES will use a subdomain of amazonses.com. The subdomain varies based on the AWS Region that you use Amazon SES in.

• **Reject message** – If the custom MAIL FROM domain's MX record is not set up correctly, Amazon SES will return a MailFromDomainNotVerified error. Emails that you attempt to send from this email address will be automatically rejected.

c. Choose Set MAIL FROM Domain. A window appears that contains the MX and SPF records that you have to add to the DNS configuration for the domain that the email address belongs to. These records use the formats shown in the following table.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>subdomain.domain.com</td>
<td>MX</td>
<td>10 feedback-smtp.region.amazonses.com</td>
</tr>
<tr>
<td>subdomain.domain.com</td>
<td>TXT</td>
<td>&quot;v=spf1 include:amazonses.com ~all&quot;</td>
</tr>
</tbody>
</table>

In the preceding records, replace `subdomain.domain.com` with your MAIL FROM subdomain, and replace `region` with the name of the AWS Region where you want to verify the MAIL FROM domain (such as us-west-2, us-east-1, or eu-west-1). Note that the value of the TXT record has to include the quotation marks.

Note these values, and then proceed to the next step.

6. Publish the DNS records to the DNS server of the custom MAIL FROM domain.

   **Important**
   To successfully set up a custom MAIL FROM domain with Amazon SES, you must publish exactly one MX record to the DNS server of your MAIL FROM domain. If the MAIL FROM domain has multiple MX records, the custom MAIL FROM setup with Amazon SES will fail.

   If Route 53 provides the DNS service for your MAIL FROM domain, and you are signed in to the AWS Management Console under the same account that you use for Route 53, then choose Publish Records Using Route 53. The DNS records are automatically applied to your domain's DNS configuration.

   If you use a different DNS provider, you have to publish the DNS records to the MAIL FROM domain's DNS server manually. The procedure for adding DNS records to your domain's DNS server varies based on your web hosting service or DNS provider.

   The procedures for publishing DNS records for your domain depend on which DNS provider you use. The following table includes links to the documentation for several common DNS providers. This list isn't a complete list of providers. If your provider isn't listed below, you can probably still set up a MAIL FROM domain. Inclusion on this list isn't an endorsement or recommendation of any company's products or services.

<table>
<thead>
<tr>
<th>DNS/Hosting Provider Name</th>
<th>Documentation Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>GoDaddy</td>
<td>• MX: Add an MX record</td>
</tr>
<tr>
<td></td>
<td>• TXT: Add a TXT record</td>
</tr>
<tr>
<td>DreamHost</td>
<td>• MX: How do I change my MX records?</td>
</tr>
<tr>
<td></td>
<td>• TXT: How do I add custom DNS records?</td>
</tr>
</tbody>
</table>
When Amazon SES detects that the records are in place, you receive an email informing you that your custom MAIL FROM domain was set up successfully. Depending on your DNS provider, there might be a delay of up to 72 hours before Amazon SES detects the MX record.

### MAIL FROM domain setup states with Amazon SES

After you configure an identity to use a custom MAIL FROM domain, the state of the setup is “pending” while Amazon SES attempts to detect the required MX record in your DNS settings. The state then changes depending on whether Amazon SES detects the MX record. The following table describes the email-sending behavior, and the Amazon SES actions associated with each state. Each time the state changes, Amazon SES sends a notification to the email address associated with your AWS account.

<table>
<thead>
<tr>
<th>State</th>
<th>Email Sending Behavior</th>
<th>Amazon SES Actions</th>
</tr>
</thead>
</table>
| Pending             | Uses custom MAIL FROM fallback setting | Amazon SES attempts to detect the required MX record for 72 hours. If unsuccessful, the state changes to “Failed”.
| Success             | Uses custom MAIL FROM domain           | Amazon SES continuously checks that the required MX record is in place.           |
| TemporaryFailure    | Uses custom MAIL FROM fallback setting | Amazon SES attempts to detect the required MX record for 72 hours. If unsuccessful, the state changes |

Setting up SPF records

An SPF record indicates to ISPs that you have authorized Amazon SES to send mail for your domain. When you use Amazon SES, your decision about whether to publish an SPF record depends on whether you only require your email to pass an SPF check by the receiving mail server, or if you want your email to comply with the additional requirements needed to pass Domain-based Message Authentication, Reporting and Conformance (DMARC) authentication based on SPF. For more information, see Authenticating Email with SPF in Amazon SES (p. 129).

Getting your SMTP credentials for Amazon SES

To use the Amazon SES SMTP interface, you must first create an SMTP user name and password. To get your SMTP Credentials, see Obtaining your Amazon SES SMTP credentials (p. 86).

**Important**
Your SMTP user name and password are not the same as your AWS access key ID and secret access key. Do not attempt to use your AWS credentials to authenticate yourself to the Amazon SES SMTP endpoint. For more information about credentials, see Types of Amazon SES credentials (p. 16).

Moving out of the Amazon SES sandbox

To help prevent fraud and abuse, and to help protect your reputation as a sender, we apply certain restrictions to new Amazon SES accounts.

We place all new accounts in the Amazon SES sandbox. While your account is in the sandbox, you can use all of the features of Amazon SES. However, when your account is in the sandbox, we apply the following restrictions to your account:

- You can only send mail to verified email addresses and domains, or to the Amazon SES mailbox simulator (p. 181).
- You can only send mail from verified email addresses and domains.

**Note**
This restriction applies even when your account isn't in the sandbox.

<table>
<thead>
<tr>
<th>State</th>
<th>Email Sending Behavior</th>
<th>Amazon SES Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed</td>
<td>Uses custom MAIL FROM fallback setting</td>
<td>Amazon SES no longer attempts to detect the required MX record. To use a custom MAIL FROM domain, you have to restart the setup process in Configuring the MAIL FROM domain (p. 66).</td>
</tr>
</tbody>
</table>
You can send a maximum of 200 messages per 24-hour period.
You can send a maximum of 1 message per second.

When your account is out of the sandbox, you can send email to any recipient, regardless of whether the recipient's address or domain is verified. However, you still have to verify all identities that you use as "From", "Source", "Sender", or "Return-Path" addresses.

Complete the procedures in this section to request that your account be removed from the sandbox.

Note
If you're using Amazon SES to send email from an Amazon EC2 instance, you might also need to request that the throttle be removed from port 25 on your Amazon EC2 instance. For more information, see How do I remove the throttle on port 25 from my EC2 instance? in the AWS Knowledge Center.

To request that your account be removed from the Amazon SES sandbox using the AWS Management Console
1. Open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, under Email Sending choose Sending Statistics.
3. For Your account details, choose Edit your account details, as shown in the following image.
4. In the account details modal, fill out the following account details:
For **Enable production access**, choose **Yes** or **No**. You can only move out of the sandbox by choosing **Yes**.

For **Mail Type**, choose the type of email that you plan to send. If more than one value applies, choose the option that applies to the majority of the email that you plan to send.
• For **Website URL**, enter the URL of your website. Providing this information helps us better understand the type of content that you plan to send.

• For **Use case description**, explain how you plan to use Amazon SES to send email. To help us process your request, you should answer the following questions:
  - How do you plan to build or acquire your mailing list?
  - How do you plan to handle bounces and complaints?
  - How can recipients opt out of receiving email from you?
  - How did you choose the sending rate or sending quota that you specified in this request?

5. For **Additional contact addresses**, tell us where you want to receive communications about your account. This can be a comma-separated list of up to 4 email addresses.

6. For **Preferred contact language**, choose whether you want to receive communications for this case in **English** or **Japanese**.

7. When you finish, choose **Submit for review**.

**Note**
Once you submit a review of your account details, you can’t edit your details until the review is complete.

Rather than submit a production access request using the AWS Management Console, you can instead submit the request using the AWS CLI. Submitting your request using the AWS CLI is helpful when you want to request production access for a large number of identities, or when you want to automate the process of setting up Amazon SES.

**To request that your account be removed from the Amazon SES sandbox using the AWS CLI**

**Note**
Before you complete the procedure in this section, you have to install and configure the AWS CLI. For more information, see the [AWS Command Line Interface User Guide](https://docs.aws.amazon.com/cli/latest/userguide/).

• At the command line, enter the following command:

```
aws sesv2 put-account-details \
--production-access-enabled \
--mail-type TRANSACTIONAL \
--website-url https://example.com \
--use-case-description "Use case description" \
--additional-contact-email-addresses info@example.com \
--contact-language EN
```

In the preceding command, do the following:

• Replace **TRANSACTIONAL** with the type of email that you plan to send through Amazon SES. You can specify either **TRANSACTIONAL** or **PROMOTIONAL**. If more than one value applies, specify the option that applies to the majority of the email that you plan to send.

• Replace **https://example.com** with the URL of your website. Providing this information helps us better understand the type of content that you plan to send.

• Replace **Use case description** with a description of how you plan to use Amazon SES to send email. To help us process your request, you should answer the following questions:
  - How do you plan to build or acquire your mailing list?
  - How do you plan to handle bounces and complaints?
  - How can recipients opt out of receiving email from you?
  - How did you choose the sending rate or sending quota that you specified in this request?
Moving out of the sandbox

- Replace info@example.com with the email addresses where you want to receive communications about your account. This can be a comma-separated list of up to 4 email addresses.
- Replace EN with your preferred language. You can specify EN for English or JP for Japanese.

**Note**
Once you submit a review of your account details, you can't edit your details until the review is complete.

The AWS Support team provides an initial response to your request within 24 hours.

In order to prevent our systems from being used to send unsolicited or malicious content, we have to consider each request carefully. If we're able to do so, we'll grant your request within this 24-hour period. However, if we need to obtain additional information from you, it might take longer to resolve your request.

We might not be able to grant your request if your use case doesn't align with our policies.

**Checking the sandbox status for your account**

You can use the Amazon SES console to determine if your account is still in the sandbox.

**To determine if your account is in the sandbox**

1. Open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, under Email Sending, choose Sending Statistics.
3. Under Your account details, you will see the status of your account.

   If your account is still being reviewed, a banner shows that your account is still under review. Your Production access status will still read as Sandbox.
The banner also reflects the status of your account should your request to move out of the sandbox be denied or have failed.

If your request to move out of the sandbox has been granted, your Production access status is noted as Enabled. This status means then your account is no longer in the sandbox in the current Region.

Your account details

<table>
<thead>
<tr>
<th>Production Access</th>
<th>Enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mail type</td>
<td>Transactional</td>
</tr>
<tr>
<td>Website URL</td>
<td><a href="https://example.com">https://example.com</a></td>
</tr>
<tr>
<td>Use case description</td>
<td>A description of how you plan to use Amazon SES</td>
</tr>
<tr>
<td>Additional contact addresses</td>
<td><a href="mailto:info@example.com">info@example.com</a></td>
</tr>
<tr>
<td>Preferred contact language</td>
<td>English</td>
</tr>
</tbody>
</table>

You can also determine whether your account is in the sandbox by sending email to an address that you haven’t verified. If your account is in the sandbox, you receive an error message stating that the destination address isn’t verified.

Configuring custom domains to handle open and click tracking

When you use event publishing (p. 288) to capture open and click events, Amazon SES makes minor changes to the emails you send. To capture open events, Amazon SES adds a 1 pixel by 1 pixel transparent image to the bottom of each email. This image has a unique file name for each email, and is hosted on a server operated by Amazon SES. To capture link click events, Amazon SES replaces the links in your emails with links to a server operated by Amazon SES. This immediately redirects the recipient to his or her intended destination. Some Amazon SES customers may want to use their own domains, rather than domains owned and operated by Amazon SES, to create a more cohesive experience for their recipients.

You can configure multiple custom domains to handle open and click tracking events. These custom domains are associated with configuration sets. When you send an email using a configuration set, if that configuration set is configured to use a custom domain, then the open and click links in that email automatically use the custom domain specified in that configuration set.

This section contains procedures for setting up a subdomain on a server you own to automatically redirect users to the open and click tracking servers operated by Amazon SES. There are two steps involved in setting up these domains. First, you configure the subdomain itself, and then you set up a
configuration set to use the custom domain. This topic contains procedures for completing both of these steps.

**Part 1: Setting up a domain for handling open and click link redirects**

The specific procedures for setting up a redirect domain vary depending on your web hosting provider (and your Content Delivery Network, if you use an HTTPS server). The procedures in the following sections provide general guidance rather than specific steps.

**Option 1: Configuring an HTTP domain**

If you plan to use an HTTP domain to handle open and click links (as opposed to an HTTPS domain), the process for configuring the subdomain involves only a few steps.

**Note**

If you set up a custom domain that uses the HTTP protocol, and you send an email that contains links that use the HTTPS protocol, your customers may see a warning message when they click the links in your email. If you plan to send emails that contain links that use the HTTPS protocol, you should use an HTTPS domain for handling open and click tracking events.

If you plan to use an HTTPS subdomain, follow the procedures in [Option 2: Configuring an HTTPS domain](p. 79) instead.

**To set up an HTTP subdomain for handling open and click links**

1. If you have not already done so, create a subdomain to use for open and click tracking links. We recommend that you create a subdomain that is specifically dedicated to handling these links.
2. Verify the subdomain for use with Amazon SES. For more information, see [Verifying domains in Amazon SES](p. 59).
3. Modify the DNS record for the subdomain. In the DNS record, add a new CNAME record that redirects requests to the Amazon SES tracking domain. The address that you redirect to depends on the AWS Region that you use Amazon SES in. The following table contains a list of tracking domains for the AWS Regions where Amazon SES is available.

<table>
<thead>
<tr>
<th>AWS Region</th>
<th>AWS tracking domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>US East (Ohio)</td>
<td>r.us-east-2.awstrack.me</td>
</tr>
<tr>
<td>US East (N. Virginia)</td>
<td>r.us-east-1.awstrack.me</td>
</tr>
<tr>
<td>US West (Oregon)</td>
<td>r.us-west-2.awstrack.me</td>
</tr>
<tr>
<td>Asia Pacific (Mumbai)</td>
<td>r.ap-south-1.awstrack.me</td>
</tr>
<tr>
<td>Asia Pacific (Seoul)</td>
<td>r.ap-northeast-2.awstrack.me</td>
</tr>
<tr>
<td>Asia Pacific (Singapore)</td>
<td>r.ap-southeast-1.awstrack.me</td>
</tr>
<tr>
<td>Asia Pacific (Sydney)</td>
<td>r.ap-southeast-2.awstrack.me</td>
</tr>
<tr>
<td>Asia Pacific (Tokyo)</td>
<td>r.ap-northeast-1.awstrack.me</td>
</tr>
<tr>
<td>Canada (Central)</td>
<td>r.ca-central-1.awstrack.me</td>
</tr>
<tr>
<td>Europe (Frankfurt)</td>
<td>r.eu-central-1.awstrack.me</td>
</tr>
<tr>
<td>Europe (Ireland)</td>
<td>r.eu-west-1.awstrack.me</td>
</tr>
</tbody>
</table>
Configuring custom open and click domains

**Option 2: Configuring an HTTPS domain**

You can also use an HTTPS domain for tracking link clicks. To set up an HTTPS domain for tracking links, you have to perform some additional steps, beyond those required for setting up an HTTP domain (p. 78).

**Note**
You can only use an HTTPS domain for tracking link clicks. Amazon SES only supports open tracking over HTTP domains.

**To set up an HTTPS subdomain for handling clicks**

1. Create a subdomain to use for click tracking links. We recommend that you create a subdomain that is specifically dedicated to handling these links.
2. Verify the subdomain for use with Amazon SES. For more information, see Verifying domains in Amazon SES (p. 59).
3. Create a new account with a Content Delivery Network (CDN), such as Amazon CloudFront.
4. Configure the CDN to forward requests to the Amazon SES tracking domain. The address that you redirect to depends on the AWS Region that you use Amazon SES in. The following table contains a list of tracking domains for the AWS Regions where Amazon SES is available.

<table>
<thead>
<tr>
<th>AWS Region</th>
<th>AWS tracking domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>US East (Ohio)</td>
<td>r.us-east-2.awstrack.me</td>
</tr>
<tr>
<td>US East (N. Virginia)</td>
<td>r.us-east-1.awstrack.me</td>
</tr>
<tr>
<td>US West (Oregon)</td>
<td>r.us-west-2.awstrack.me</td>
</tr>
<tr>
<td>Asia Pacific (Mumbai)</td>
<td>r.ap-south-1.awstrack.me</td>
</tr>
<tr>
<td>Asia Pacific (Seoul)</td>
<td>r.ap-northeast-2.awstrack.me</td>
</tr>
<tr>
<td>Asia Pacific (Singapore)</td>
<td>r.ap-southeast-1.awstrack.me</td>
</tr>
<tr>
<td>Asia Pacific (Sydney)</td>
<td>r.ap-southeast-2.awstrack.me</td>
</tr>
<tr>
<td>Asia Pacific (Tokyo)</td>
<td>r.ap-northeast-1.awstrack.me</td>
</tr>
<tr>
<td>Canada (Central)</td>
<td>r.ca-central-1.awstrack.me</td>
</tr>
<tr>
<td>Europe (Frankfurt)</td>
<td>r.eu-central-1.awstrack.me</td>
</tr>
<tr>
<td>Europe (Ireland)</td>
<td>r.eu-west-1.awstrack.me</td>
</tr>
</tbody>
</table>

**Note**
Depending on your web hosting provider, it may take several minutes for the changes you make to the subdomain’s DNS record to take effect. Your web hosting provider or IT organization can provide additional information about these delays.
Configuring custom open and click domains

<table>
<thead>
<tr>
<th>AWS Region</th>
<th>AWS tracking domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Europe (London)</td>
<td>r.eu-west-2.awstrack.me</td>
</tr>
<tr>
<td>South America (São Paulo)</td>
<td>r.sa-east-1.awstrack.me</td>
</tr>
<tr>
<td>AWS GovCloud (US)</td>
<td>r.us-gov-west-1.awstrack.me</td>
</tr>
</tbody>
</table>

5. If you use Amazon CloudFront as your CDN, complete the following procedures:
   a. On the CloudFront Distributions page, choose the distribution that corresponds with your CDN.
   b. On the Behaviors tab, choose the default behavior, and then choose Edit.
   c. For Cache Based on Selected Request Headers, choose All.
   d. For Query String Forwarding and Caching, choose Forward all, cache based on all.
   e. Add an alternate domain name to your distribution. The subdomain that you use has to be verified in Amazon SES. For more information, see Configuring Alternate Domain Names and HTTPS in the Amazon CloudFront Developer Guide.

If you use a CDN other than CloudFront, you might need to complete similar steps. For more information, refer to the documentation for your CDN.

6. If you use Route 53 to manage the DNS configuration for your domain and CloudFront as your CDN, create an Alias record in Route 53 that refers to your CloudFront distribution (such as d111111abcdef8.cloudfront.net). For more information, see Creating Records by Using the Amazon Route 53 Console in the Amazon Route 53 Developer Guide.

Otherwise, in the DNS configuration for your subdomain, add a CNAME record that refers to the address of your CDN.

7. Acquire an SSL certificate from a trusted Certificate Authority. The certificate should cover both the subdomain you created in step 1 as well as the CDN you configured in steps 3–5. Upload the certificate to the CDN.

Part 2: Setting up a configuration set to refer to a custom open and click tracking domain

After you configure your domain to handle open and click tracking redirects, you must set up an event destination in a configuration set to refer to your custom domain. You can complete this step using the Amazon SES console or the CreateConfigurationSetTrackingOptions API operation. This section contains procedures for completing these tasks using the Amazon SES console; for more information about using the API, see CreateConfigurationSetTrackingOptions in the Amazon Simple Email Service API Reference.

To create a new configuration set event destination that refers to a custom tracking domain

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation bar on the left side of the screen, choose Configuration Sets.
3. Choose Create Configuration Set.
4. For Configuration Set Name, type a name for the configuration set, and then choose Create Configuration Set.
5. In the list of configuration sets, select the box next to the configuration set you created in the previous step. On the Actions menu, choose Edit.
6. On the Event Destinations tab, for Add Destination, choose an event destination type. For more information about the options in this menu, see Step 2: Add event destination (p. 291).
7. For **Event types**, choose either **Click**, **Open**, or both, depending on the types of events you want to track.

8. For **Domain**, choose **Use your own subdomain**.

9. For **Select a verified domain**, choose the domain that you want to use for open and click event tracking. In the text field to the left of the menu, you can optionally specify a subdomain of the parent domain.

10. Configure the remaining options as you normally would. For more information about setting up event destinations, see **Step 2: Add event destination (p. 291)**.

11. Choose **Save**.

---

### Setting up VPC endpoints with Amazon SES

Many Amazon SES customers have corporate policies in place that limit the ability of their internal systems to connect to the public internet. These policies prevent these customers from using the public Amazon SES endpoints.

To work within these restrictions, you can use Amazon Virtual Private Cloud (Amazon VPC). With Amazon VPC, you can deploy AWS resources into a virtual network that exists in an isolated area of the AWS Cloud. For more information about Amazon VPC, see the Amazon VPC User Guide.

To use Amazon SES with Amazon VPC, you first have to create an Amazon EC2 instance in your organization's VPC. You can then connect to this instance and use it to send email through Amazon SES. This section contains instructions for configuring your Amazon EC2 instance and creating an Amazon VPC endpoint for Amazon SES.

**Note**

Amazon Elastic Compute Cloud (Amazon EC2) restricts email traffic over port 25 by default. To avoid timeouts when sending email through the SMTP endpoint from Amazon EC2, you can request that these restrictions be removed. For more information, see How do I remove the restriction on port 25 from my Amazon EC2 instance or AWS Lambda function? in the AWS Knowledge Center. Alternatively, you can use a different port (such as 587 or 2587) to avoid this issue.

### Prerequisites

Before you complete the procedure in this section, you have to complete the following steps:

- Create a Virtual Private Cloud. For procedures, see [Getting started with IPv4 for Amazon VPC](#).
- Launch an Amazon EC2 instance in your VPC. For more information, see [Launching an EC2 instance into your default VPC](#).

### Setting up Amazon SES in Amazon VPC

The process of setting up a VPC endpoint to use with Amazon SES consists of a few separate steps. First, you have to identify the private IP address of the Amazon EC2 instance that you want to use with the VPC endpoint. Next, you create a security group that allows the instance to communicate with SMTP ports. After that, you create a VPC endpoint for Amazon SES. Finally, you test the connection to the VPC endpoint to ensure that it's configured properly.

#### Step 1: Find the Private IP Address of Your Amazon EC2 Instance

To set up an Amazon EC2 instance to use an Amazon SES VPC endpoint, you first have to find the private IP of the instance. You use this IP address in a later step.
To find the private IP of an Amazon EC2 instance

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, under Instances, choose Instances.
3. In the list of Amazon EC2 instances, choose the instance that you want to use to connect to the VPC endpoint.
4. In the detail pane at the bottom of the screen, on the Description tab, copy the IP address next to Private IP.

Step 2: Create the Security Group

In Amazon EC2, a security group lets you control inbound and outbound communications to and from your VPC. In this step, you create a security group that lets the Amazon EC2 instance communicate with SMTP endpoints.

To create the security group

1. In the navigation pane of the Amazon EC2 console, under Network & Security, choose Security Groups.
2. Choose Create security group.
3. Under Basic details, do the following:
   - For Security group name, enter a unique name that identifies the security group.
   - For Description, enter some text that describes the purpose of the security group.
   - For VPC, choose the VPC that you want to use Amazon SES in.

When you finish, the Basic details section resembles the example in the following image.

4. Under Inbound rules, choose Add rule.
5. Under Inbound rule 1, do the following:
   - For Type, choose Custom TCP.
• For **Port range**, enter the port number that you want to use to send email. You can use any of the following port numbers: **25, 465, 587, 2465, or 2587**.

• For **Source type**, choose **Custom**.

• For **Source**, enter the private IP of your Amazon EC2 instance (that is, the address that you found earlier).

6. (Optional) If you want to add an inbound rule for additional ports, choose **Add rule** again. Then, repeat the preceding step to add additional ports. You can create rules for any or all of the port numbers listed in the preceding step.

7. When you finish, choose **Create security group**.

**Step 3: Create the VPC endpoint**

In Amazon VPC, a **VPC endpoint** lets you connect your VPC to supported AWS services. In this case, you configure Amazon VPC so that your Amazon EC2 security group can connect to Amazon SES.

**To create the VPC endpoint**

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.

2. Under **Virtual Private Cloud**, choose **Endpoints**.

3. Choose **Create Endpoint**.

4. On the **Create Endpoint** page, for **Service category**, choose **AWS services**.

5. Under **Service Name**, use the search box to search for "email", as shown in the following image.

   ![Service Name](image)

Choose the **email-smtp** service for your current AWS Region.

6. For **VPC**, choose the Virtual Private Cloud that you want to use.

7. Under **Security group**, choose the security group that you created earlier, as shown in the following image.

83
8. Choose **Create endpoint**. Wait approximately 5 minutes while Amazon VPC creates the endpoint. When the endpoint is ready to use, the value in the **Status** column changes to "available", as shown in the following image.

---

**Step 4: Test the connection to the VPC endpoint**

When you complete the process of configuring the VPC endpoint, you should test the connection to ensure that the VPC endpoint is configured properly. You can test the connection by using command-line tools that are included with most operating systems.

**To test the connection to the VPC endpoint**

1. Connect to your Amazon EC2 instance.

   For information about connecting to Linux instances, see [Connect to your Linux instance](#) in the **Amazon EC2 User Guide for Linux Instances**.

   For information about connecting to Windows instances, see [Getting started](#) in the **Amazon EC2 User Guide for Windows Instances**.

2. Send a test email by completing the procedure in [Using the command line to send email using the Amazon SES SMTP interface](#) (p. 105).
Email sending methods in Amazon SES

You can send an email with Amazon Simple Email Service (Amazon SES) using the Amazon SES console, the Amazon SES Simple Mail Transfer Protocol (SMTP) interface, or the Amazon SES API. You typically use the console to send test emails and manage your sending activity. To send bulk emails, you use either the SMTP interface or the API. For information about Amazon SES email pricing, see Amazon SES Pricing.

- If you want to use an SMTP-enabled software package, application, or programming language to send email through Amazon SES, or integrate Amazon SES with your existing mail server, use the Amazon SES SMTP interface. For more information, see Using the Amazon SES SMTP interface to send email (p. 85).
- If you want to call Amazon SES by using raw HTTP requests, use the Amazon SES API. For more information, see Using the Amazon SES API to send email (p. 109).

Before you send emails, see Setting up email with Amazon SES (p. 46).

**Important**
When you send an email to multiple recipients (recipients are "To", "CC", and "BCC" addresses) and the call to Amazon SES fails, the entire email is rejected and none of the recipients will receive the intended email. We therefore recommend that you send an email to one recipient at a time.

Using the Amazon SES SMTP interface to send email

To send production email through Amazon SES, you can use the Simple Mail Transfer Protocol (SMTP) interface or the Amazon SES API. For more information about the Amazon SES API, see Using the Amazon SES API to send email (p. 109). This section describes the SMTP interface.

Amazon SES sends email using SMTP, which is the most common email protocol on the internet. You can send email through Amazon SES by using a variety of SMTP-enabled programming languages and software to connect to the Amazon SES SMTP interface. This section explains how to get your Amazon SES SMTP credentials, how to send email by using the SMTP interface, and how to configure several pieces of software and mail servers to use Amazon SES for email sending.

**Note**
For solutions to common problems that you might encounter when you use Amazon SES through its SMTP interface, see Amazon SES SMTP issues (p. 479).

To send email using the Amazon SES SMTP interface, you need the following:

- An AWS account. For more information, see Signing up for AWS (p. 46).
- The SMTP endpoint address. For a list of Amazon SES SMTP endpoints, see Connecting to an Amazon SES SMTP endpoint (p. 90).
- The SMTP interface port number. The port number varies with the connection method. For more information, see Connecting to an Amazon SES SMTP endpoint (p. 90).
- An SMTP user name and password. SMTP credentials are unique to each AWS Region. If you plan to use the SMTP interface to send email in multiple AWS Regions, you need a username and password for each Region.
Important

Your SMTP user name and password aren't identical to your AWS access keys or the credentials that you use to sign in to the Amazon SES console. For information about how to generate your SMTP user name and password, see Obtaining your Amazon SES SMTP credentials (p. 86).

- Client software that can communicate using Transport Layer Security (TLS). For more information, see Connecting to an Amazon SES SMTP endpoint (p. 90).
- An email address that you've verified with Amazon SES. For more information, see Verifying identities in Amazon SES (p. 47).
- Increased sending quotas, if you want to send large quantities of email. For more information, see Managing your Amazon SES sending quotas (p. 144).

Then, you can send email by doing the following:

- To configure SMTP-enabled software to send email through the Amazon SES SMTP interface, see Sending email through Amazon SES using software packages (p. 91).
- To program an application to send email through Amazon SES, see Sending email through Amazon SES from your application (p. 93).
- To configure your existing email server to send all of your outgoing mail through Amazon SES, see Integrating Amazon SES with your existing email server (p. 93).
- To interact with the Amazon SES SMTP interface using the command line, which can be useful for testing, see Test your connection to the Amazon SES SMTP interface using the command line (p. 103).

For a list of SMTP response codes, see SMTP response codes returned by Amazon SES (p. 480).

Email information to provide

When you access Amazon SES through the SMTP interface, your SMTP client application assembles the message, so the information you need to provide depends on the application that you're using. At a minimum, the SMTP exchange between a client and a server requires a source address, a destination address, and message data.

If you're using the SMTP interface and have feedback forwarding enabled, then your bounces, complaints, and delivery notifications are sent to the "MAIL FROM" address. Any "Reply-To" address that you specify isn't used.

Obtaining your Amazon SES SMTP credentials

You need an Amazon SES SMTP user name and password to access the Amazon SES SMTP interface.

The credentials that you use to send email through the Amazon SES SMTP interface are unique to each AWS Region. If you use the Amazon SES SMTP interface to send email in more than one Region, you must generate a set of SMTP credentials (p. 86) for each Region that you plan to use.

Your SMTP password is different from your AWS secret access key. For more information about credentials, see Types of Amazon SES credentials (p. 16).

Obtaining Amazon SES SMTP credentials using the Amazon SES console

When you generate SMTP credentials by using the Amazon SES console, the Amazon SES console creates an IAM user with the appropriate policies to call Amazon SES and provides you with the SMTP credentials associated with that user.
Requirement

An IAM user can create Amazon SES SMTP credentials, but the IAM user's policy must give them permission to use IAM itself, because Amazon SES SMTP credentials are created by using IAM. Your IAM policy must allow you to perform the following IAM actions: `iam:ListUsers`, `iam:CreateUser`, `iam:CreateAccessKey`, and `iam:PutUserPolicy`. If you try to create Amazon SES SMTP credentials using the console and your IAM user doesn't have these permissions, you see an error that states that your account is "not authorized to perform iam:ListUsers."

To create your SMTP credentials

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, choose SMTP Settings.
3. In the content pane, choose Create My SMTP Credentials.
4. For Create User for SMTP, type a name for your SMTP user. Alternatively, you can use the default value that is provided in this field. When you finish, choose Create.
   
   ![Create User for SMTP](image)

5. Choose Show User SMTP Credentials. Your SMTP credentials are shown on the screen. Download or copy these credentials and store them in a safe place, as you cannot view or save your credentials after you dismiss this dialog box.
   
   ![Show User SMTP Security Credentials](image)

6. Choose Close Window.

You can view a list of existing SMTP credentials that you've created using this procedure by going to the IAM console at https://console.aws.amazon.com/iam/. In the navigation pane, under Access management, choose Users. Use the search bar to find all users that you've assigned SMTP credentials.

You can also use the IAM console to delete existing SMTP users. To learn more about deleting users, see https://docs.aws.amazon.com/IAM/latest/UserGuide/Managing-IAM-Users in the IAM Getting Started Guide.

If you want to rotate your SMTP credentials, complete the procedure above to generate a new set of SMTP credentials. Then, test the new credentials to ensure that they work as expected. Finally, delete
the IAM user associated with the old SMTP credentials in the IAM console. For more information about deleting users in IAM, see Managing users in the IAM User Guide.

### Obtaining Amazon SES SMTP credentials by converting existing AWS credentials

If you have an IAM user that you set up using the IAM interface, you can derive the user's Amazon SES SMTP credentials from their AWS credentials.

**Important**

Don't use temporary AWS credentials to derive SMTP credentials. The Amazon SES SMTP interface doesn't support SMTP credentials that have been generated from temporary security credentials.

To enable the IAM user to send email using the Amazon SES SMTP interface, you need to do the following two steps:

- Derive the user's SMTP credentials from their AWS credentials using the algorithm provided in this section. Because you are starting from AWS credentials, the SMTP user name is the same as the AWS access key ID, so you just need to generate the SMTP password.
- Apply the following policy to the IAM user:

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ses:SendRawEmail",
      "Resource": "*"
    }
  ]
}
```

For more information about using Amazon SES with IAM, see Controlling access to Amazon SES (p. 397).

**Note**

Although you can generate Amazon SES SMTP credentials for any IAM user, we recommend that you create a separate IAM user when you generate your SMTP credentials. For information about why it is good practice to create users for specific purposes, go to IAM Best Practices.

The following pseudocode shows an algorithm that converts an AWS secret access key to an Amazon SES SMTP password.

```plaintext
// Modify this variable to include your AWS secret access key
key = "wJalrXUtnFEMI/K7MDENG/bPxRfYiZVXrQ==";

// Modify this variable to refer to the AWS Region that you want to use to send email.
region = "us-west-2";

// The values of the following variables should always stay the same.
date = "1111111111";
service = "ses";
terminal = "aws4_request";
message = "SendRawEmail";
version = 0x04;

kDate = HmacSha256(date, "AWS4" + key);
kRegion = HmacSha256(region, kDate);
kService = HmacSha256(service, kRegion);
kTerminal = HmacSha256(terminal, kTerminal);
```
Some programming languages include libraries that you can use to convert an IAM secret access key into an SMTP password. This section includes a code example that you can use to convert an AWS secret access key to an Amazon SES SMTP password using Python.

**Note**
The following example uses *f-strings* that were introduced in Python 3.6; if using an older version, they won't work.
Currently, the Python SDK (Boto3) officially supports 2.7 and 3.6 (or later). However, 2.7 support is deprecated and will be dropped on 7/15/2021, so you'll need to upgrade to at least 3.6.

```python
#!/usr/bin/env python3
import hmac
import hashlib
import base64
import argparse

SMTP_REGIONS = [
    'us-east-2',       # US East (Ohio)
    'us-east-1',       # US East (N. Virginia)
    'us-west-2',       # US West (Oregon)
    'ap-south-1',      # Asia Pacific (Mumbai)
    'ap-northeast-2',  # Asia Pacific (Seoul)
    'ap-southeast-1',  # Asia Pacific (Singapore)
    'ap-southeast-2',  # Asia Pacific (Sydney)
    'ap-northeast-1',  # Asia Pacific (Tokyo)
    'ca-central-1',    # Canada (Central)
    'eu-central-1',    # Europe (Frankfurt)
    'eu-west-1',       # Europe (Ireland)
    'eu-west-2',       # Europe (London)
    'sa-east-1',       # South America (Sao Paulo)
    'us-gov-west-1',   # AWS GovCloud (US)
]

# These values are required to calculate the signature. Do not change them.
DATE = "11111111"
SERVICE = "ses"
MESSAGE = "SendRawEmail"
TERMINAL = "aws4_request"
VERSION = 0x04

def sign(key, msg):
    return hmac.new(key, msg.encode('utf-8'), hashlib.sha256).digest()

def calculate_key(secret_access_key, region):
    if region not in SMTP_REGIONS:
        raise ValueError(f"The {region} Region doesn't have an SMTP endpoint.")
    signature = sign(("AWS4" + secret_access_key).encode('utf-8'), DATE)
    signature = sign(signature, region)
    signature = sign(signature, SERVICE)
    signature = sign(signature, TERMINAL)
    signature = sign(signature, MESSAGE)
    signature_and_version = bytes([VERSION]) + signature
    smtp_password = base64.b64encode(signature_and_version)
    return smtp_password.decode('utf-8')
```
```python
def main():
    parser = argparse.ArgumentParser(
        description='Convert a Secret Access Key for an IAM user to an SMTP password.')
    parser.add_argument(
        'secret', help='The Secret Access Key to convert.')
    parser.add_argument(
        'region', help='The AWS Region where the SMTP password will be used.',
        choices=SMTP_REGIONS)
    args = parser.parse_args()
    print(calculate_key(args.secret, args.region))

if __name__ == '__main__':
    main()
```

To obtain your SMTP password by using this script, save the preceding code as `smtp_credentials_generate.py`. Then, at the command line, execute the following command:

```
python path/to/smtp_credentials_generate.py wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY us-east-1
```

In the preceding command, do the following:

- Replace `path/to/` with the path to the location where you saved `smtp_credentials_generate.py`.
- Replace `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY` with the secret access key that you want to convert into an SMTP password.
- Replace `us-east-1` with the AWS Region in which you want to use the SMTP credentials.

When this script runs successfully, the only output is your SMTP password.

To use this script, first save the preceding code as `smtp_credentials_generate.py`. Then, at the command line, execute the following command:

```
python path/to/smtp_credentials_generate.py wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY us-east-1
```

In the preceding command, do the following:

- Replace `path/to/` with the path to the location where you saved `smtp_credentials_generate.py`.
- Replace `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY` with the Secret Access Key that you want to convert into an SMTP password.
- Replace `us-east-1` with the AWS Region in which you want to use the SMTP credentials.

When this script runs successfully, the only output is your SMTP password.

**Connecting to an Amazon SES SMTP endpoint**

To send email using the Amazon SES SMTP interface, you connect to an SMTP endpoint. For a complete list of Amazon SES SMTP endpoints, see [Amazon Simple Email Service endpoints and quotas](https://docs.aws.amazon.com/AmazonSES/latest/DeveloperGuide/smtp-list.html) in the AWS General Reference.
The Amazon SES SMTP endpoint requires that all connections be encrypted using Transport Layer Security (TLS). (Note that TLS is often referred to by the name of its predecessor protocol, SSL.) Amazon SES supports two mechanisms for establishing a TLS-encrypted connection: STARTTLS and TLS Wrapper. Check the documentation for your software to determine whether it supports STARTTLS, TLS Wrapper, or both.

**Note**

Amazon Elastic Compute Cloud (Amazon EC2) restricts email traffic over port 25 by default. To avoid timeouts when sending email through the SMTP endpoint from Amazon EC2, you can request that these restrictions be removed. For more information, see How do I remove the restriction on port 25 from my Amazon EC2 instance or AWS Lambda function? in the AWS Knowledge Center.

Alternatively, you can use a different port, or use an Amazon VPC endpoint (p. 81).

### STARTTLS

STARTTLS is a means of upgrading an unencrypted connection to an encrypted connection. There are versions of STARTTLS for a variety of protocols; the SMTP version is defined in RFC 3207.

To set up a STARTTLS connection, the SMTP client connects to the Amazon SES SMTP endpoint on port 25, 587, or 2587, issues an EHLO command, and waits for the server to announce that it supports the STARTTLS SMTP extension. The client then issues the STARTTLS command, initiating TLS negotiation. When negotiation is complete, the client issues an EHLO command over the new encrypted connection, and the SMTP session proceeds normally.

### TLS Wrapper

TLS Wrapper (also known as SMTPS or the Handshake Protocol) is a means of initiating an encrypted connection without first establishing an unencrypted connection. With TLS Wrapper, the Amazon SES SMTP endpoint does not perform TLS negotiation: it is the client's responsibility to connect to the endpoint using TLS, and to continue using TLS for the entire conversation. TLS Wrapper is an older protocol, but many clients still support it.

To set up a TLS Wrapper connection, the SMTP client connects to the Amazon SES SMTP endpoint on port 465 or 2465. The server presents its certificate, the client issues an EHLO command, and the SMTP session proceeds normally.

### Sending email through Amazon SES using software packages

There are a number of commercial and open source software packages that support sending email via SMTP. Here are some examples:

- Blogging platforms
- RSS aggregators
- List management software
- Workflow systems

You can configure any such SMTP-enabled software to send email through the Amazon SES SMTP interface. For instructions on how to configure SMTP for a particular software package, see the documentation for that software.

The following procedure shows how to set up Amazon SES sending in JIRA, a popular issue-tracking solution. With this configuration, JIRA can notify users via email whenever there is a change in the status of a software issue.

**To Configure JIRA to Send Email Using Amazon SES**

1. Using your web browser, log in to JIRA with administrator credentials.
2. In the browser window, choose Administration.
3. On the System menu, choose Mail.
4. On the Mail administration page, choose Mail Servers.
5. Choose Configure new SMTP mail server.
6. On the Add SMTP Mail Server form, fill in the following fields:
   a. Name—A descriptive name for this server.
   b. From address—The address from which email will be sent. You will need to verify this email
      address with Amazon SES before you can send from it. For more information about verification,
      see Verifying identities in Amazon SES (p. 47).
   c. Email prefix—A string that JIRA prepends to each subject line prior to sending.
   d. Protocol—Choose SMTP.
      
      Note
      
      If you cannot connect to Amazon SES using this setting, try SECURE_SMTP.
   e. Host Name—See Connecting to an Amazon SES SMTP endpoint (p. 90) for a list of Amazon
      SES SMTP endpoints. For example, if you want to use the Amazon SES endpoint in the US West
      (Oregon) region, the host name would be email-smtp.us-west-2.amazonaws.com.
   f. SMTP Port—25, 587, or 2587 (to connect using STARTTLS), or 465 or 2465 (to connect using
      TLS Wrapper).
   g. TLS—Select this check box.
   h. Username—Your SMTP username.
   i. Password—Your SMTP password.

   Settings for TLS Wrapper are shown below.

7. Choose Test Connection. If the test email that JIRA sends through Amazon SES arrives successfully,
   then your configuration is complete.
Sending email through Amazon SES from your application

Many programming languages support sending email using SMTP. This capability might be built into the programming language itself, or it might be available as an add-on, plug-in, or library. You can take advantage of this capability by sending email through Amazon SES from within application programs that you write.

For examples in C# and Java, see Send an email by accessing the Amazon SES SMTP interface programmatically (p. 21) in the Getting Started section.

Integrating Amazon SES with your existing email server

If you currently administer your own email server, you can use the Amazon SES SMTP endpoint to send all of your outgoing email to Amazon SES. There is no need to modify your existing email clients and applications; the changeover to Amazon SES will be transparent to them.

Several mail transfer agents (MTAs) support sending email through SMTP relays. This section provides general guidance on how to configure some popular MTAs to send email using Amazon SES SMTP interface.

The Amazon SES SMTP endpoint requires that all connections be encrypted using Transport Layer Security (TLS).

Topics

- Integrating Amazon SES with Postfix (p. 93)
- Integrating Amazon SES with Sendmail (p. 97)
- Integrating Amazon SES with Microsoft Windows Server IIS SMTP (p. 100)
- Integrating Amazon SES with Exim (p. 101)

Integrating Amazon SES with Postfix

Postfix is an alternative to the widely used Sendmail Message Transfer Agent (MTA). For information about Postfix, go to http://www.postfix.org. The procedures in this topic will work with Linux, macOS, or Unix.

Note

Postfix is a third-party application, and isn't developed or supported by Amazon Web Services. The procedures in this section are provided for informational purposes only, and are subject to change without notice.

Prerequisites

Before you complete the procedures in this section, you have to perform the following tasks:

- Uninstall Sendmail, if it's already installed on your system. The procedure for completing this step varies depending on the operating system you use.
- Install Postfix. The procedure for completing this step varies depending on the operating system you use.
- Install a SASL authentication package. The procedure for completing this step varies depending on the operating system you use. For example, if you use a RedHat-based system, you should install the cyrus-sasl-plain package. If you use a Debian- or Ubuntu-based system, you should install the libsasl2-modules package.
- Verify an email address or domain to use for sending email. For more information, see Verifying email addresses in Amazon SES (p. 47).
- If your account is still in the sandbox, you can only send email to verified email addresses. For more information, see Moving out of the Amazon SES sandbox (p. 71).
Configuring Postfix

Complete the following procedures to configure your mail server to send email through Amazon SES using Postfix.

To configure Postfix

1. At the command line, type the following command:

   ```bash
   sudo postconf -e "relayhost = [email-smtp.us-west-2.amazonaws.com]:587" \
   "smtp_sasl_auth_enable = yes" \
   "smtp_sasl_security_options = noanonymous" \
   "smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd" \
   "smtp_use_tls = yes" \
   "smtp_tls_security_level = encrypt" \
   "smtp_tls_note_starttls_offer = yes"
   ```

   **Note**
   If you use Amazon SES in an AWS Region other than US West (Oregon), replace `email-smtp.us-west-2.amazonaws.com` in the preceding command with the SMTP endpoint of the appropriate region. For more information, see Regions (p. 457).

2. In a text editor, open the file `/etc/postfix/master.cf`. Search for the following entry:

   ```
   -o smtp_fallback_relay=
   ```

   If you find this entry, comment it out by placing a # (hash) character at the beginning of the line.

   Save and close the file.

   Otherwise, if this entry isn't present, proceed to the next step.

3. In a text editor, open the file `/etc/postfix/sasl_passwd`. If the file doesn't already exist, create it.

4. Add the following line to `/etc/postfix/sasl_passwd`:

   ```
   [email-smtp.us-west-2.amazonaws.com]:587 SMTPUSERNAME:SMTPPASSWORD
   ```

   **Note**
   Replace `SMTPUSERNAME` and `SMTPPASSWORD` with your SMTP username and password, respectively. Your SMTP user name and password aren't the same as your AWS access key ID and secret access key. For more information about credentials, see the section called "Obtaining your SMTP credentials" (p. 86).

   If you use Amazon SES in an AWS Region other than US West (Oregon), replace `email-smtp.us-west-2.amazonaws.com` in the example above with the SMTP endpoint of the appropriate region. For more information, see Regions (p. 457).

   Save and close `sasl_passwd`.

5. At a command prompt, type the following command to create a hashmap database file containing your SMTP credentials:

   ```bash
   sudo postmap hash:/etc/postfix/sasl_passwd
   ```

6. *(Optional)* The `/etc/postfix/sasl_passwd` and `/etc/postfix/sasl_passwd.db` files you created in the previous steps aren't encrypted. Because these files contain your SMTP credentials, we recommend that you modify the files' ownership and permissions in order to restrict access to them. To restrict access to these files:

   a. At a command prompt, type the following command to change the ownership of the files:
Using the SMTP interface

7. Tell Postfix where to find the CA certificate (needed to verify the Amazon SES server certificate). The command you use in this step varies based on your operating system.

   a. If you use Amazon Linux, Red Hat Enterprise Linux, or a related distribution, type the following command:

   ```
   sudo postconf -e 'smtp_tls_CAfile = /etc/ssl/certs/ca-bundle.crt'
   ```

   b. If you use Ubuntu or a related distribution, type the following command:

   ```
   sudo postconf -e 'smtp_tls_CAfile = /etc/ssl/certs/ca-certificates.crt'
   ```

   c. If you use macOS, you can generate the certificate from your system keychain. To generate the certificate, type the following command at the command line:

   ```
   sudo security find-certificate -a -p /System/Library/Keychains/SystemRootCertificates.keychain > /etc/ssl/certs/ca-bundle.crt
   ```

   After you generate the certificate, type the following command:

   ```
   sudo postconf -e 'smtp_tls_CAfile = /etc/ssl/certs/ca-bundle.crt'
   ```

8. Type the following command to start the Postfix server (or to reload the configuration settings if the server is already running):

   ```
   sudo postfix start; sudo postfix reload
   ```

9. Send a test email by typing the following at a command line, pressing Enter after each line. Replace `sender@example.com` with your From email address. The From address has to be verified for use with Amazon SES. Replace `recipient@example.com` with the destination address. If your account is still in the sandbox, the recipient address also has to be verified. Finally, the final line of the message has to contain a single period (.) with no other content.

   ```
   sendmail -f sender@example.com recipient@example.com
   From: Sender Name <sender@example.com>
   Subject: Amazon SES Test
   This message was sent using Amazon SES.
   .
   ```

10. Check the mailbox associated with the recipient address. If the email doesn't arrive, check your junk mail folder. If you still can't locate the email, check the mail log on the system that you used to send the email (typically located at `/var/log/maillog`) for more information.

**Advanced usage example**

This example shows how to send an email that uses a configuration set (p. 249), and that uses MIME-multipart encoding to send both a plain text and an HTML version of the message, along with an attachment. It also includes a link tag (p. 508), which can be used for categorizing click events. The
content of the email is specified in an external file, so that you do not have to manually type the commands in the Postfix session.

To send a multipart MIME email using Postfix

1. In a text editor, create a new file called `mime-email.txt`.
2. In the text file, paste the following content, replacing the values in red with the appropriate values for your account:

   ```plaintext
   X-SES-CONFIGURATION-SET: ConfigSet
   From: Sender Name <sender@example.com>
   Subject: Amazon SES Test
   MIME-Version: 1.0
   Content-Type: multipart/mixed; boundary="YWVhZDFlY2QzMGQ2N2U0YTZmODU"
   --YWVhZDFlY2QzMGQ2N2U0YTZmODU
   Content-Type: multipart/alternative; boundary="3NjM0N2QwMTE4MWQ0ZTg2NTYxZQ"
   --3NjM0N2QwMTE4MWQ0ZTg2NTYxZQ
   Content-Type: text/plain; charset=UTF-8
   Content-Transfer-Encoding: quoted-printable
   Amazon SES Test
   This message was sent from Amazon SES using the SMTP interface.
   For more information, see:
   --3NjM0N2QwMTE4MWQ0ZTg2NTYxZQ
   Content-Type: text/html; charset=UTF-8
   Content-Transfer-Encoding: quoted-printable
   <html>
   <head>
   </head>
   <body>
   <h1>Amazon SES Test</h1>
   This message was sent from Amazon SES using the SMTP interface.</p>
   For more information, see:
   <a href="http://docs.aws.amazon.com/ses/latest/DeveloperGuide/send-email-smtp.html">Using the Amazon SES SMTP Interface to Send Email</a>
   in the <em>Amazon SES Developer Guide</em>.</p>
   </body>
   </html>
   --3NjM0N2QwMTE4MWQ0ZTg2NTYxZQ--
   Content-Type: application/octet-stream
   MIME-Version: 1.0
   Content-Transfer-Encoding: base64
   Content-Disposition: attachment; filename="customers.txt"
   SUQsRmlyc3ROYW1iLENvdW50cnkKMsQ4LEpvaG4sU3RpbGVzLENh
   bmFkYQo5MjM4OSxKawUsTGl1LENoaW5hCjczNjIxTglLc2h5LjMwMmVlZjMyZDQ4MDc=
   bm10ZWhgU3RhdGVzCjI4OTMaW5hWEtXl1lbmdhcixJbmRyYQ==
   --YWVhZDFlY2QzMGQ2N2U0YTZmODU--
   
   Save and close the file.
3. At the command line, type the following command. Replace `sender@example.com` with your email address, and replace `recipient@example.com` with the recipient's email address.
sendmail -f sender@example.com recipient@example.com < mime-email.txt

If the command runs successfully, it exits without providing any output.

4. Check your inbox for the email. If the message wasn't delivered, check your system's mail log.

Integrating Amazon SES with Sendmail

Sendmail was released in the early 1980s, and has been continuously improved ever since. It is a flexible and configurable message transfer agent (MTA) with a large community of users. Sendmail was acquired by Proofpoint in 2013, but Proofpoint continues to offer an open source version of Sendmail. You can download the open source version of Sendmail from the Proofpoint website, or through the package managers of most Linux distributions.

The procedure in this section shows you how to configure Sendmail to send email through Amazon SES. This procedure was tested on a server running Ubuntu 18.04.2 LTS.

Note
Sendmail is a third-party application, and isn't developed or supported by Amazon Web Services. The procedures in this section are provided for informational purposes only, and are subject to change without notice.

Prerequisites

Before you complete the procedure in this section, you should complete the following steps:

- Install the Sendmail package on your server.

  Note
  Depending on which operating system distribution you use, you might also need to install the following packages: sendmail-cf, m4, and cyrus-sasl-plain.

- Verify an identity to use as your From address. For more information, see Verifying email addresses in Amazon SES (p. 47)

  If your account is in the Amazon SES sandbox, you also have to verify the addresses that you send email to. For more information, see Moving out of the Amazon SES sandbox (p. 71).

If you're using Amazon SES to send email from an Amazon EC2 instance, you should also complete the following steps:

- If you’re using Amazon SES to send email from an Amazon EC2 instance, you might need to assign an Elastic IP Address to your Amazon EC2 instance in order for receiving email providers to accept your email. For more information, see Amazon EC2 Elastic IP addresses in the Amazon EC2 User Guide for Linux Instances.

- Amazon Elastic Compute Cloud (Amazon EC2) restricts email traffic over port 25 by default. To avoid timeouts when sending email through the SMTP endpoint from Amazon EC2, you can request that these restrictions be removed. For more information, see How do I remove the restriction on port 25 from my Amazon EC2 instance or AWS Lambda function? in the AWS Knowledge Center.

  Alternatively, you can modify the procedure in this section to use port 587 rather than port 25.

Configuring Sendmail

Complete the steps in this section to configure Sendmail to send email by using Amazon SES.
Important
The procedure in this section assumes that you want to use Amazon SES in the US West (Oregon) AWS Region. If you want to use a different Region, replace all instances of email-smtp.us-west-2.amazonaws.com in this procedure with the SMTP endpoint of the desired region. For a list of SMTP endpoint URLs for the AWS Regions where Amazon SES is available, see Amazon Simple Email Service (Amazon SES) in the AWS General Reference.

To configure Sendmail

1. In a file editor, open the file /etc/mail/authinfo. If the file doesn't exist, create it.

   Add the following line to /etc/mail/authinfo:

   ```plaintext
   ```

   In the preceding example, make the following changes:
   - Replace email-smtp.us-west-2.amazonaws.com with the Amazon SES SMTP endpoint that you want to use.
   - Replace smtpUsername with your Amazon SES SMTP username.
   - Replace smtpPassword with your Amazon SES SMTP password.

   Note
   Your SMTP username and password are different from your AWS Access Key ID and Secret Access Key. For more information about obtaining your SMTP username and password, see Obtaining your Amazon SES SMTP credentials (p. 86).

   When you finish, save authinfo.

2. At the command line, enter the following command to generate the /etc/mail/authinfo.db file:

   ```bash
   sudo sh -c 'makemap hash /etc/mail/authinfo.db < /etc/mail/authinfo'
   ```

3. At the command line, type the following command to add support for relaying to the Amazon SES SMTP endpoint.

   ```bash
   sudo sh -c 'echo "Connect:email-smtp.us-west-2.amazonaws.com RELAY" >> /etc/mail/access'
   ```

   In the preceding command, replace email-smtp.us-west-2.amazonaws.com with the address of the Amazon SES SMTP endpoint that you want to use.

4. At the command line, type the following command to regenerate /etc/mail/access.db:

   ```bash
   sudo sh -c 'makemap hash /etc/mail/access.db < /etc/mail/access'
   ```

5. At the command line, type the following command to create backups of the sendmail.cf and sendmail.mc files:

   ```bash
   sudo sh -c 'cp /etc/mail/sendmail.cf /etc/mail/sendmail_cf.backup && cp /etc/mail/sendmail.mc /etc/mail/sendmail_mc.backup'
   ```

6. Add the following lines to the /etc/mail/sendmail.mc file before any MAILER() definitions.

   ```plaintext
   define(`SMART_HOST', `email-smtp.us-west-2.amazonaws.com')dnl
   define(`RELAY_MAILER_ARGS', `TCP $h 25')dnl
   ```
define(`confAUTH_MECHANISMS', `LOGIN PLAIN')
dnl
FEATURE(`authinfo', `hash -o /etc/mail/authinfo.db')
dnl
MASQUERADE_AS(`example.com')
dnl
FEATURE(masquerade_envelope)
dnl
FEATURE(masquerade_entire_domain)
dnl

In the preceding text, do the following:

- Replace `email-smtp.us-west-2.amazonaws.com` with the Amazon SES SMTP endpoint that you want to use.
- Replace `example.com` with the domain that you want to use to send email.

When you finish, save the file.

**Note**
Amazon EC2 restricts communications over port 25 by default. If you’re using Sendmail in an Amazon EC2 instance, you should complete the Request to Remove Email Sending Limitations.

7. At the command line, type the following command to make `sendmail.cf` writeable:

```
sudo chmod 666 /etc/mail/sendmail.cf
```

8. At the command line, type the following command to regenerate `sendmail.cf`:

```
sudo sh -c 'm4 /etc/mail/sendmail.mc > /etc/mail/sendmail.cf'
```

**Note**
If you encounter errors such as "Command not found" and "No such file or directory," make sure that the `m4` and `sendmail-cf` packages are installed on your system.

9. At the command line, type the following command to reset the permissions of `sendmail.cf` to read only:

```
sudo chmod 644 /etc/mail/sendmail.cf
```

10. At the command line, type the following command to restart Sendmail:

```
sudo /etc/init.d/sendmail restart
```

11. Complete the following steps to send a test email:
   a. At the command line, enter the following command.

   ```
   /usr/sbin/sendmail -vf sender@example.com recipient@example.com
   ```

   Replace `sender@example.com` with your From email address. Replace `recipient@example.com` with the To address. When you finish, press Enter.

   b. Enter the following message content. Press Enter at the end of each line.

   ```
   From: sender@example.com
   To: recipient@example.com
   Subject: Amazon SES test email
   This is a test message sent from Amazon SES using Sendmail.
   ```

   When you finish entering the content of the email, press Ctrl+D to send it.
12. Check the recipient email's client for the email. If you can't find the email, check the junk mail folder. If you still can't find the email, check the Sendmail log on your mail server. The log is often located at /var/log/mail.log or /var/log/maillog.

**Integrating Amazon SES with Microsoft Windows Server IIS SMTP**

You can configure Microsoft Windows Server's IIS SMTP server to send email through Amazon SES. These instructions were written using Microsoft Windows Server 2012 on an Amazon EC2 instance. You can use the same configuration on Microsoft Windows Server 2008 and Microsoft Windows Server 2008 R2.

*Note*  
Windows Server is a third-party application, and isn't developed or supported by Amazon Web Services. The procedures in this section are provided for informational purposes only, and are subject to change without notice.

**To integrate the Microsoft Windows Server IIS SMTP server with Amazon SES**

1. First, set up Microsoft Windows Server 2012 using the following instructions.
   
   a. From the Amazon EC2 management console, launch a new Microsoft Windows Server 2012 Base Amazon EC2 instance.
   
   b. Connect to the instance and log into it using Remote Desktop by following the instructions in Getting Started with Amazon EC2 Windows Instances.
   
   c. Launch the Server Manager Dashboard.
   
   d. Install the Web Server role. Be sure to include the IIS 6 Management Compatibility tools (an option under the Web Server checkbox).
   
   e. Install the SMTP Server feature.

2. Next, configure the IIS SMTP service using the following instructions.
   
   a. Return to the Server Manager Dashboard.
   
   b. From the Tools menu, choose Internet Information Services (IIS) 6.0 Manager.
   
   c. Right-click SMTP Virtual Server #1 and then select Properties.
   
   d. On the Access tab, under Relay Restrictions, choose Relay.
   
   e. In the Relay Restrictions dialog box, choose Add.
   
   f. Under Single Computer, enter 127.0.0.1 for the IP address. You have now granted access for this server to relay email to Amazon SES through the IIS SMTP service.

   In this procedure, we assume that your emails are generated on this server. If the application that generates the email runs on a separate server, you need to grant relaying access for that server in IIS SMTP.

   *Note*  
   To extend the SMTP relay to private subnets, for Relay Restriction, use Single Computer 127.0.0.1 and Group of Computers 172.1.1.0 - 255.255.255.0 (in the netmask section). For Connection, use Single Computer 127.0.0.1 and Group of Computers 172.1.1.0 - 255.255.255.0 (in the netmask section).

3. Finally, configure the server to send email through Amazon SES using the following instructions.
   
   a. Return to the SMTP Virtual Server #1 Properties dialog box and then choose the Delivery tab.
   
   
   c. Select Basic Authentication and then enter your Amazon SES SMTP username and password. You can obtain these credentials from the Amazon SES console using the procedure in Obtaining your Amazon SES SMTP credentials (p. 86).
Important
Your SMTP user name and password are not the same as your AWS access key ID and secret access key. Do not attempt to use your AWS credentials to authenticate yourself against the SMTP endpoint. For more information about credentials, see Types of Amazon SES credentials (p. 16).

d. Ensure that TLS encryption is selected.
e. Return to the Delivery tab.
f. Choose Outbound Connections.
g. In the Outbound Connections dialog box, ensure that the port is 25 or 587.

Note
If your mail server is hosted on Amazon EC2, and you plan to send email using port 25, you must first remove the port 25 restrictions for your account. For more information, see How do I remove the restriction on port 25 from my Amazon EC2 instance or AWS Lambda function? in the AWS Knowledge Center.

h. Choose Advanced.
i. For the Smart host name, enter the Amazon SES endpoint that you will use (for example, email-smtp.us-west-2.amazonaws.com). For a list of endpoint URLs for the AWS Regions where Amazon SES is available, see Amazon Simple Email Service (Amazon SES) in the AWS General Reference.
j. Return to the Server Manager Dashboard.
k. On the Server Manager Dashboard, right-click SMTP Virtual Server #1 and then restart the service to pick up the new configuration.
l. Send an email through this server. You can examine the message headers to confirm that it was delivered through Amazon SES.

Integrating Amazon SES with Exim

Exim is a Mail Transfer Agent (MTA) that is highly flexible and configurable. To learn more about Exim, visit the Exim website.

Note
Exim is a third-party application, and isn't developed or supported by Amazon Web Services. The procedures in this section are provided for informational purposes only, and are subject to change without notice.

To configure Exim to send email through Amazon SES

1. In a text editor, open the file /etc/exim.conf.local. If the file doesn't exist, copy the template from /etc/exim4/exim4.conf.template.
2. In /etc/exim.conf.local, make the following changes:
   a. In the routers section, after the begin routers line, add the following:

   ```plaintext
   send_via_ses:
   driver = manualroute
   domains = ! +local_domains
   transport = ses_smtp
   route_list = * email-smtp.us-west-2.amazonaws.com;
   ```

   In the preceding code, replace email-smtp.us-west-2.amazonaws.com with the SMTP endpoint that you want to use to send the message. For more information, see Regions and Amazon SES (p. 457).
   b. In the transports section, after the begin transports line, add the following:
ses_smtp:
driver = smtp
port = 587
hosts_require_auth = *
hosts_require_tls = *

c. In the authenticators section, after the `begin authenticators` line, add the following:

```
ses_login:
driver = plaintext
public_name = LOGIN
client_send = : USERNAME : PASSWORD
```

In the preceding code, replace `USERNAME` with your SMTP username, and `PASSWORD` with your SMTP password.

**Important**
Your SMTP credentials are not the same as your AWS Access Key ID and Secret Access Key. For information about obtaining your SMTP credentials, see Obtaining your Amazon SES SMTP credentials (p. 86).

3. Save `/etc/exim.conf.local`.

4. When you finish updating the configuration, enter the following command to restart Exim.

```
sudo /etc/init.d/exim4 restart
```

**Note**
This command might differ depending on which operating system you use.

5. At the command line, complete the following steps to send a test message:

a. Enter the following command:

```
exim -v recipient@example.com
```

In the preceding command, replace `recipient@example.com` with the address that you want to send the message to.

b. Enter the following, pressing `Enter` at the end of each line:

```
From: sender@example.com
Subject: Test message
This is a test.
```

In the preceding command, replace `sender@example.com` with the address that you want to send the message from.

When you press `Enter` after the final period (.), Exim begins the conversation with the SMTP server. If the connection remains open after the message is sent, press `Ctrl+D` to close it.

**Tip**
If the message isn’t delivered, check your system’s mail log for errors. The Exim mail log is usually located at `/var/log/exim4/mainlog`.
Test your connection to the Amazon SES SMTP interface using the command line

You can interact with the Amazon SES SMTP interface from your operating system's command line. The methods described in this section are intended to be used to test your connection to the Amazon SES SMTP endpoint, validate your SMTP credentials, and troubleshoot connection issues. These procedures use tools and libraries that are included with most common operating systems.

For additional information about troubleshooting SMTP connection problems, see Amazon SES SMTP issues (p. 479).

Prerequisites

When you connect to the Amazon SES SMTP interface, you have to provide a set of SMTP credentials. These SMTP credentials are different from your standard AWS credentials. The two types of credentials aren't interchangeable. For more information about obtaining your SMTP credentials, see the section called "Obtaining your SMTP credentials" (p. 86).

Testing your connection to the Amazon SES SMTP interface

You can use the command line to test your connection to the Amazon SES SMTP interface without authenticating or sending any messages. This procedure is useful for troubleshooting basic connectivity issues.

This section includes procedures for testing your connection using both OpenSSL (which is included with most Linux, macOS, and Unix distributions, and is also available for Windows) and the Test-NetConnection cmdlet in PowerShell (which is included with most recent versions of Windows).

Linux, macOS, or Unix

There are two ways to connect to the Amazon SES SMTP interface with OpenSSL: using explicit SSL over port 587, or using implicit SSL over port 465.

To connect to the SMTP interface using explicit SSL

- At the command line, enter the following command to connect to the Amazon SES SMTP server:

```bash
openssl s_client -crlf -quiet -starttls smtp -connect email-smtp.us-west-2.amazonaws.com:587
```

In the preceding command, replace `email-smtp.us-west-2.amazonaws.com` with the URL of the Amazon SES SMTP endpoint for your AWS Region. For more information, see Regions (p. 457).

If the connection was successful, you see output similar to the following:

```
depth=2 C = US, O = Amazon, CN = Amazon Root CA 1
verify return:1
```

```bash
depth=1 C = US, O = Amazon, OU = Server CA 1B, CN = Amazon
verify return:1
depth=0 CN = email-smtp.us-west-2.amazonaws.com
verify return:1
250 Ok
```

The connection automatically closes after about 10 seconds of inactivity.

Alternatively, you can use implicit SSL to connect to the SMTP interface over port 465.
To connect to the SMTP interface using implicit SSL

- At the command line, enter the following command to connect to the Amazon SES SMTP server:

```
openssl s_client -crlf -quiet -connect email-smtp.us-west-2.amazonaws.com:465
```

In the preceding command, replace `email-smtp.us-west-2.amazonaws.com` with the URL of the Amazon SES SMTP endpoint for your AWS Region. For more information, see Regions (p. 457).

If the connection was successful, you see output similar to the following:

```
depth=2 C = US, O = Amazon, CN = Amazon Root CA 1
verify return:1
depth=1 C = US, O = Amazon, OU = Server CA 1B, CN = Amazon
verify return:1
depth=0 CN = email-smtp.us-west-2.amazonaws.com
verify return:1
220 email-smtp.amazonaws.com ESMTP SimpleEmailService-d-VCSHDP1YZ
```

The connection automatically closes after about 10 seconds of inactivity.

PowerShell

You can use the `Test-NetConnection` cmdlet in PowerShell to connect to the Amazon SES SMTP server.

**Note**
The `Test-NetConnection` cmdlet can determine whether your computer can connect to the Amazon SES SMTP endpoint. However, it doesn't test whether your computer can make an implicit or explicit SSL connection to the SMTP endpoint. To test an SSL connection, you can either install OpenSSL for Windows, or complete the procedure in Using the command line to send email using the Amazon SES SMTP interface (p. 105) to send a test email.

To connect to the SMTP interface using the `Test-NetConnection` cmdlet

- In PowerShell, enter the following command to connect to the Amazon SES SMTP server:

```
Test-NetConnection -Port 587 -ComputerName email-smtp.us-west-2.amazonaws.com
```

In the preceding command, replace `email-smtp.us-west-2.amazonaws.com` with the URL of the Amazon SES SMTP endpoint for your AWS Region, and replace `587` with the port number. For more information about regional endpoints in Amazon SES, see Regions (p. 457).

If the connection was successful, you see output that resembles the following example:

```
ComputerName : email-smtp.us-west-2.amazonaws.com
RemoteAddress : 198.51.100.126
RemotePort : 587
InterfaceAlias : Ethernet
SourceAddress : 203.0.113.46
TcpTestSucceeded : True
```
Using the command line to send email using the Amazon SES SMTP interface

You can also use the command line to send messages using the Amazon SES SMTP interface. This procedure is useful for testing SMTP credentials and for testing the ability of specific recipients to receive messages that you send by using Amazon SES.

### Linux, macOS, or Unix

When an email sender connects to an SMTP server, the client issues a standard set of requests, and the server replies to each request with a standard response. This series of requests and responses is called an **SMTP conversation**. When you connect to the Amazon SES SMTP server using OpenSSL, the server expects an SMTP conversation to occur.

When you use OpenSSL to connect to the SMTP interface, you have to encode your SMTP credentials using base64 encoding. This section includes procedures for encoding your credentials using base64.

#### To send an email from the command line using the SMTP interface

1. At the command line, enter the following command to encode your SMTP user name, replacing `SMTPUsername` with your SMTP user name:

   ```bash
   echo -n "SMTPUsername" | openssl enc -base64
   ```

   Make a note of the output of this command.

2. At the command line, enter the following command to encode your SMTP password, replacing `SMTPPassword` with your SMTP password:

   ```bash
   echo -n "SMTPPassword" | openssl enc -base64
   ```

   Make a note of the output of this command.

3. In a text editor, create a new file. Paste the following code into the file:

   ```plaintext
   EHLO example.com
   AUTH LOGIN
   Base64EncodedSMTPUserName
   Base64EncodedSMTPPassword
   MAIL FROM: sender@example.com
   RCPT TO: recipient@example.com
   DATA
   X-SES-CONFIGURATION-SET: ConfigSet
   From: Sender Name <sender@example.com>
   To: recipient@example.com
   Subject: Amazon SES SMTP Test
   This message was sent using the Amazon SES SMTP interface.
   .
   QUIT
   ```

4. Make the following changes to the file that you created in the previous step:

   - Replace `example.com` with your sending domain.
   - Replace `Base64EncodedSMTPUserName` with your base64-encoded SMTP user name.
   - Replace `Base64EncodedSMTPPassword` with your base64-encoded SMTP password.
   - Replace `sender@example.com` with the email address you are sending from. This identity must be verified.
   - Replace `recipient@example.com` with the destination email address. If your Amazon SES account is still in the sandbox, this address must be verified.
• Replace ConfigSet with the name of the configuration set (p. 249) that you want to use when you send this email.

  **Note**
  If you don’t want to use a configuration set, you can omit the entire line that begins with X-SES-CONFIGURATION-SET.

When you finish, save the file as input.txt.

5. At the command line, choose one of the following options:

• **To send using explicit SSL over port 587** – Enter the following command:

  ```bash
  openssl s_client -crlf -quiet -starttls smtp -connect email-smtp.us-west-2.amazonaws.com:587 < input.txt
  ```

• **To send using implicit SSL over port 465** – Enter the following command:

  ```bash
  openssl s_client -crlf -quiet -connect email-smtp.us-west-2.amazonaws.com:465 < input.txt
  ```

  **Note**
  Replace email-smtp.us-west-2.amazonaws.com with the URL of the Amazon SES SMTP endpoint for your AWS Region. For more information, see **Regions** (p. 457).

If the message was accepted by Amazon SES, you see output that resembles the following example:

```
250 Ok 01010160d7de98d8-21e57d9a-JZho-416c-bbe1-8ebaAexample-000000
```

The string of numbers and text that follows 250 Ok is the message ID of the email.

  **Note**
  The connection closes automatically after about 10 seconds of inactivity.

**PowerShell**

You can use the **Net.Mail.SmtpClient** class to send email using explicit SSL over port 587.

  **Note**
  The Net.Mail.SmtpClient class is officially obsolete, and Microsoft recommends that you use third-party libraries. This code is intended for testing purposes only, and shouldn’t be used for production workloads.

**To send an email through PowerShell using explicit SSL**

1. In a text editor, create a new file. Paste the following code into the file:

```powershell
function SendEmail($Server, $Port, $Sender, $Recipient, $Subject, $Body) {
    $Credentials = [Net.NetworkCredential](Get-Credential)
    $SMTPClient = New-Object Net.Mail.SmtpClient($Server, $Port)
    $SMTPClient.EnableSsl = $true
    $SMTPClient.Credentials = New-Object
    System.Net.NetworkCredential($Credentials.Username, $Credentials.Password);
    try {
        Write-Output "Sending message..."
```
> $SMTPClient.Send($Sender, $Recipient, $Subject, $Body)
> Write-Output "Message successfully sent to $Recipient"
> } catch [System.Exception] {
> Write-Output "An error occurred:"
> Write-Error $_
> }
>
> function SendTestEmail(){
> $Server = "email-smtp.us-west-2.amazonaws.com"
> $Port = 587
> $Subject = "Test email sent from Amazon SES"
> $Body = "This message was sent from Amazon SES using PowerShell (explicit SSL, port 587)."
> $Sender = "sender@example.com"
> $Recipient = "recipient@example.com"
> SendEmail $Server $Port $Sender $Recipient $Subject $Body
> }
> SendTestEmail
>
> When you finish, save the file as SendEmail.ps1.

2. Make the following changes to the file that you created in the previous step:
   - Replace sender@example.com with the email address that you want to send the message from.
   - Replace recipient@example.com with the email address that you want to send the message to.
   - Replace email-smtp.us-west-2.amazonaws.com with the URL of the Amazon SES SMTP endpoint for your AWS Region. For more information, see Regions and Amazon SES (p. 457).

3. In PowerShell, enter the following command:

   .\path\to\SendEmail.ps1

   In the preceding command, replace path\to\SendEmail.ps1 with the path to the file that you created in step 1.

4. When prompted, enter your SMTP user name and password.

Alternatively, you can use the System.Web.Mail.SmtpMail class to send email using implicit SSL over port 465.

**Note**
The System.Web.Mail.SmtpMail class is officially obsolete, and Microsoft recommends that you use third-party libraries. This code is intended for testing purposes only, and shouldn't be used for production workloads.

To send an email through PowerShell using implicit SSL

1. In a text editor, create a new file. Paste the following code into the file:

   ```powershell
   [System.Reflection.Assembly]:LoadWithPartialName("System.Web") > $null
   function SendEmail($Server, $Port, $Sender, $Recipient, $Subject, $Body) {
   $Credentials = [Net.NetworkCredential](Get-Credential)
   }```

   ```powershell
   When you finish, save the file as SendEmail.ps1.
   ```
```powershell
$mail.From = $Sender
$mail.To = $Recipient
$mail.Subject = $Subject
$mail.Body = $Body
try {
    Write-Output "Sending message..."
    [System.Web.Mail.SmtpMail]::Send($mail)
    Write-Output "Message successfully sent to $($Recipient)"
} catch [System.Exception] {
    Write-Output "An error occurred:"
    Write-Error $_
}
```

```powershell
function SendTestEmail(){
    $Server = "email-smtp.us-west-2.amazonaws.com"
    $Port = 465
    $Subject = "Test email sent from Amazon SES"
    $Body = "This message was sent from Amazon SES using PowerShell (implicit SSL, port 465)."
    $Sender = "sender@example.com"
    $Recipient = "recipient@example.com"
    SendEmail $Server $Port $Sender $Recipient $Subject $Body
}
SendTestEmail
```

When you finish, save the file as SendEmail.ps1.

2. Make the following changes to the file that you created in the previous step:
   - Replace `sender@example.com` with the email address that you want to send the message from.
   - Replace `recipient@example.com` with the email address that you want to send the message to.
   - Replace `email-smtp.us-west-2.amazonaws.com` with the URL of the Amazon SES SMTP endpoint for your AWS Region. For more information, see Regions and Amazon SES (p. 457).

3. In PowerShell, enter the following command:
   ```powershell
   .\path\to\SendEmail.ps1
   ```
In the preceding command, replace `path\to\SendEmail.ps1` with the path to the file that you created in step 1.

4. When prompted, enter your SMTP user name and password.

**Using the Amazon SES API to send email**

To send production email through Amazon SES, you can use the Simple Mail Transfer Protocol (SMTP) interface or the Amazon SES API. For more information about the SMTP interface, see Using the Amazon SES SMTP interface to send email (p. 85). This section describes how to send email by using the API.

You can also use the Amazon SES API to send email. When you send an email using the API, you specify the content of the message, and Amazon SES assembles a MIME email for you. Alternatively, you can assemble the email yourself so that you have complete control over the content of the message. For more information about the API, see the Amazon Simple Email Service API Reference. For a list of endpoint URLs for the AWS Regions where Amazon SES is available, see Amazon Simple Email Service endpoints and quotas in the AWS General Reference.

You can call the API in the following ways:

- **Make direct HTTPS requests**—This is the most advanced method, because you have to manually handle authentication and signing of your requests, and then manually construct the requests. For information about how to make requests, see Using the Amazon SES API (p. 407).

- **Use an AWS SDK**—AWS SDKs make it easy to access the APIs for several AWS services, including Amazon SES. When you use an SDK, it takes care of authentication, request signing, retry logic, error handling, and other low-level functions so that you can focus on building applications that delight your customers.

- **Use a command line interface**—The AWS Command Line Interface is the command line tool for Amazon SES. We also offer the AWS Tools for Windows PowerShell for those who script in the PowerShell environment.

Regardless of whether you access the Amazon SES API directly or indirectly through an AWS SDK, the AWS Command Line Interface or the AWS Tools for Windows PowerShell, the Amazon SES API provides two different ways for you to send an email, depending on how much control you want over the composition of the email message:

- **Formatted**—Amazon SES composes and sends a properly formatted email message. You need only supply "From:" and "To:" addresses, a subject, and a message body. Amazon SES takes care of all the rest. For more information, see Sending formatted email using the Amazon SES API (p. 109).

- **Raw**—You manually compose and send an email message, specifying your own email headers and MIME types. If you are experienced in formatting your own email, the raw interface gives you more control over the composition of your message. For more information, see Sending raw email using the Amazon SES API (p. 110).

**Topics in this section:**

- Sending formatted email using the Amazon SES API (p. 109)
- Sending raw email using the Amazon SES API (p. 110)

**Sending formatted email using the Amazon SES API**

You can send a formatted email by using the AWS Management Console or by calling the Amazon SES API through an application directly, or indirectly through an AWS SDK, the AWS Command Line Interface, or the AWS Tools for Windows PowerShell.
The Amazon SES API provides the `SendEmail` action, which lets you compose and send a formatted email. `SendEmail` requires a `From:` address, `To:` address, message subject, and message body—text, HTML, or both. For a complete description of `SendEmail`, go to the Amazon Simple Email Service API Reference.

**Note**
The email address string must be 7-bit ASCII. If you want to send to or from email addresses that contain unicode characters in the domain part of an address, you must encode the domain using Punycode. For more information, see RFC 3492.

For an example of how to compose a formatted message using the AWS SDK for Java or the AWS SDK for .NET, see Send an email using the AWS SDK for Java (p. 34) or Send an email using the AWS SDK for .NET (p. 31), respectively.

For tips on how to increase your email sending speed when you make multiple calls to `SendEmail`, see Increasing throughput with Amazon SES (p. 478).

**Sending raw email using the Amazon SES API**

You can use the Amazon SES `SendRawEmail` operation to send highly customized messages to your recipients.

This section includes procedures for constructing and sending raw email using the Amazon SES API.

**About email header fields**

Simple Mail Transfer Protocol (SMTP) specifies how email messages are to be sent by defining the mail envelope and some of its parameters, but it does not concern itself with the content of the message. Instead, the Internet Message Format (RFC 5322) defines how the message is to be constructed.

With the Internet Message Format specification, every email message consists of a header and a body. The header consists of message metadata, and the body contains the message itself. For more information about email headers and bodies, see Email format and Amazon SES (p. 14).

**Using MIME**

The SMTP protocol was originally designed to send email messages that only contained 7-bit ASCII characters. This specification makes SMTP insufficient for non-ASCII text encodings (such as Unicode), binary content, or attachments. The Multipurpose Internet Mail Extensions standard (MIME) was developed to make it possible to send many other kinds of content using SMTP.

The MIME standard works by breaking the message body into multiple parts and then specifying what is to be done with each part. For example, one part of an email message body might be plain text, while another might be HTML. In addition, MIME allows email messages to contain one or more attachments. Message recipients can view the attachments from within their email clients, or they can save the attachments.

The message header and content are separated by a blank line. Each part of the email is separated by a boundary, a string of characters that denotes the beginning and ending of each part.

The multipart message in the following example contains a text and an HTML part. It also contains an attachment.

```
From: "Sender Name" <sender@example.com>
To: recipient@example.com
Subject: Customer service contact info
Content-Type: multipart/mixed;
    boundary="a3f166a86b56ff6c37755292d690675717ea3cd9de81228ec2b76ed4a15d6d1a"

--a3f166a86b56ff6c37755292d690675717ea3cd9de81228ec2b76ed4a15d6d1a
```
The content type for the message is `multipart/mixed`, which indicates that the message has many parts (in this example, a body and an attachment), and the receiving client must handle each part separately. Nested within the body section is a second part that uses the `multipart/alternative` content type. This content type indicates that each part contains alternative versions of the same content (in this case, a text version and an HTML version). If the recipient's email client can display HTML content, then it shows the HTML version of the message body. If the recipient's email client can't display HTML content, then it shows the plain text version of the message body. Both versions of the message also contain an attachment (in this case, a short text file that contains some customer names).

When you nest a MIME part within another part, as in this example, the nested part must use a boundary parameter that is distinct from the boundary parameter in the parent part. These boundaries should be unique strings of characters. To define a boundary between MIME parts, type two hyphens (--) followed by the boundary string. At the end of a MIME part, place two hyphens at both the beginning and the end of the boundary string.

**MIME Encoding**

To maintain compatibility with older systems, Amazon SES honors the 7-bit ASCII limitation of SMTP as defined in [RFC 2821](https://tools.ietf.org/html/rfc2821). If you want to send content that contains non-ASCII characters, you must encode those characters into a format that uses 7-bit ASCII characters.

**Email addresses**

To encode an email address that is used in the message envelope, use Punycode encoding.

For example, to send an email to `##@example.com`, use Punycode encoding on the local part of the address (the part before the `@` sign). The resulting, encoded address is `xn--cpay30b@example.com`. 
Note
This rule only applies to email addresses that you specify in the message envelope, not the message headers. When you use the SendRawEmail API, the addresses you specify in the Source and Destinations parameters define the envelope sender and recipients, respectively.

For more information about Punycode encoding, see RFC 3492.

Email headers

To encode a message header, use MIME encoded-word syntax. MIME encoded word syntax uses the following format:

```
=?charset?encoding?encoded-text?=
```

The value of `encoding` can be either Q or B. If the value of encoding is Q, then the value of `encoded-text` has to use Q-encoding. If the value of encoding is B, then the value of `encoded-text` has to use base64 encoding.

For example, if you want to use the string "Як ти поживаєш?" in the subject line of an email, you can use either of the following encodings:

- **Q-encoding**
  ```
  =?utf-8?Q?=D0=AF=D0=BA_=D1=82=D0=B8_=D0=BF=D0=BE=D0=B6=D0=B8=D0=B2=D0=B0=D1=94=D1=88=3F?=
  ```

- **Base64 encoding**
  ```
  =?utf-8?B?0K/QuiDRgtC4INC/0L7QttC40LLQsNGU0Yg/?=
  ```

For more information about Q-encoding, see RFC 2047. For more information about base64 encoding, see RFC 2045.

Message body

To encode the body of a message, you can use quoted-printable encoding or base64 encoding. Then, use the Content-Transfer-Encoding header to indicate which encoding scheme you used.

For example, assume the body of your message contains the following text:

```plaintext
१९७२ मे रे टॉमलिन्सन ने पहला ई-मेल भेजा। रे टॉमलिन्सन ने ही सूचयांकन @ चैनल का चयन किया और इसी को ईमेल का आवश्यक माना जाता है.
```

If you choose to encode this text using base64 encoding, first specify the following header:

```
Content-Transfer-Encoding: base64
```

Then, in the body section of the email, include the base64-encoded text:

```
4KWn4KWv4KWt4KWoIOCkrucIhvyDgpLDgpYcg4KSf4KWJ4KSu4KSy4KS/4KSC4KSm4KSoIOChkDCl
byDgpKrgbLmgpgLlgpL4g4KSIleCkrucIh+CkmiDgpLjgpLlgpgLbpgYfppLg4KSt4KWW4KSc4KSm4K
Ihng4KW4KWHICKn+CliCeCkrucIhvuCkvp+CkugCkuOCkqCkqDgpLjgpYcg4KSm4KWAIOCkucIcuOCkECl
jeCktvCkqiCkriCkpeCkr1bAIOCkmucIkv+CkqDCljeCkriCkpuDgpLXjgpL4g4KSm4KSm4KSoIOChk
leCkvp+Ckr+CkviDgpLttgpLAg4KSH4KSo4KNNg4KSm4KWAIOCktvCkriDgpLjgpLg4KgpLtrg4KSV
4KSm4KSc4KSm4KSo4Knh4KSm4KWAIOCktvCkriDgpLjgpLg4KgpLtrg4KSV
```

112
**Note**
In some cases, you can use the 8bit Content-Transfer-Encoding in messages that you send using Amazon SES. However, if Amazon SES has to make any changes to your messages (for example, when you use open and click tracking (p. 506)), 8-bit-encoded content might not appear correctly when it arrives in recipients' inboxes. For this reason, you should always encode content that isn't 7-bit ASCII.

**File attachments**

To attach a file to an email, you have to encode the attachment using base64 encoding. Attachments are typically placed in dedicated MIME message parts, which include the following headers:

- **Content-Type:** The file type of the attachment. The following are examples of common MIME Content-Type declarations:
  - **Plain text file:** Content-Type: text/plain; name="sample.txt"
  - **Microsoft Word Document:** Content-Type: application/msword; name="document.docx"
  - **JPG image:** Content-Type: image/jpeg; name="photo.jpeg"
- **Content-Disposition:** Specifies how the recipient's email client should handle the content. For attachments, this value is Content-Disposition: attachment.
- **Content-Transfer-Encoding:** The scheme that was used to encode the attachment. For file attachments, this value is almost always base64.

Amazon SES accepts most common file types. For a list of file types that Amazon SES doesn't accept, see [Unsupported attachment types](#) (p. 513).

**Sending raw email using the Amazon SES API**

The Amazon SES API provides the `SendRawEmail` action, which lets you compose and send an email message in the format that you specify. For a complete description of `SendRawEmail`, see the [Amazon Simple Email Service API Reference](#).

**Note**

For tips on how to increase your email sending speed when you make multiple calls to `SendRawEmail`, see [Increasing throughput with Amazon SES](#) (p. 478).

The message body must contain a properly formatted, raw email message, with appropriate header fields and message body encoding. Although it is possible to construct the raw message manually within an application, it is much easier to do so using existing mail libraries.

**Java**

The following code example shows how to use the [JavaMail](#) library and the [AWS SDK for Java](#) to compose and send a raw email.

```java
package com.amazonaws.samples;
import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.io.PrintStream;
import java.nio.ByteBuffer;
import java.util.Properties;

// JavaMail libraries. Download the JavaMail API
// from https://javaee.github.io/javamail/
import javax.activation.DataHandler;
import javax.activation.DataSource;
import javax.activation.FileDataSource;
import javax.mail.Message;

public class Example {
    public static void main(String[] args) throws IOException {
        // Create a Properties object to store the server settings
        Properties properties = System.getProperties();

        // Set the email server
        properties.put("mail.smtp.host", "smtp.example.com");
        properties.put("mail.smtp.port", "587");
        properties.put("mail.smtp.auth", "true");

        // Create a session object
       javax.mail.Session session = javax.mail.Session.getDefaultSession(
                javax.mail.Session.AUTO_RETRIEVE);

        // Create a message
        javax.mail.Message message = session.createMessage(
                javax.mail.Message.ROUTINE_MESSAGE);

        // Set the envelope for the message
        message.setFrom("you@example.com");
        message.setRecipient(javax.mail.Message.RECIPIENT_TO, new DataHandler("sample.txt"));
        message.setSubject("Attached File");

        // Attach a file to the message
        DataSource fileDataSource = new FileDataSource("sample.txt");
        message.setAttachment(fileDataSource);

        // Send the message
        session.sendMessage(message);
    }
}
```
import javax.mail.MessagingException;
import javax.mail.Session;
import javax.mail.internet.AddressException;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeBodyPart;
import javax.mail.internet.MimeMessage;
import javax.mail.internet.MimeMultipart;

// AWS SDK libraries. Download the AWS SDK for Java
// from https://aws.amazon.com/sdk-for-java
import com.amazonaws.regions.Regions;
import com.amazonaws.services.simpleemail.AmazonSimpleEmailService;
import com.amazonaws.services.simpleemail.AmazonSimpleEmailServiceClientBuilder;
import com.amazonaws.services.simpleemail.model.RawMessage;
import com.amazonaws.services.simpleemail.model.SendRawEmailRequest;

public class AmazonSESSample {

    // Replace sender@example.com with your "From" address.
    // This address must be verified with Amazon SES.
    private static String SENDER = "Sender Name <sender@example.com>";

    // Replace recipient@example.com with a "To" address. If your account
    // is still in the sandbox, this address must be verified.
    private static String RECIPIENT = "recipient@example.com";

    // Specify a configuration set. If you do not want to use a configuration
    // set, comment the following variable, and the
    // ConfigurationSetName=CONFIGURATION_SET argument below.
    private static String CONFIGURATION_SET = "ConfigSet";

    // The subject line for the email.
    private static String SUBJECT = "Customer service contact info";

    // The full path to the file that will be attached to the email.
    // If you're using Windows, escape backslashes as shown in this variable.
    private static String ATTACHMENT = "C:\Users\sender\customers-to-contact.xlsx";

    // The email body for recipients with non-HTML email clients.
    private static String BODY_TEXT = "Hello,
    + "Please see the attached file for a list "
    + "of customers to contact."

    // The HTML body of the email.
    private static String BODY_HTML = "<html>
    + "<head></head>"
    + "<body>"
    + "<h1>Hello!</h1>
    + "<p>Please see the attached file for a "
    + "list of customers to contact.</p>
    + "</body>
    + "</html>";

    public static void main(String[] args) throws AddressException, MessagingException,
            IOException {
        Session session = Session.getDefaultInstance(new Properties());

        // Create a new MimeMessage object.
        MimeMessage message = new MimeMessage(session);

        // Add subject, from and to lines.
        message.setSubject(SUBJECT, "UTF-8");
        message.setFrom(new InternetAddress(SENDER));
        message.setRecipients(Message.RecipientType.TO,
                InternetAddress.parse(RECIPIENT));
// Create a multipart/alternative child container.
MimeMultipart msg_body = new MimeMultipart("alternative");

// Create a wrapper for the HTML and text parts.
MimeBodyPart wrap = new MimeBodyPart();

// Define the text part.
MimeBodyPart textPart = new MimeBodyPart();
textPart.setContent(BODY_TEXT, "text/plain; charset=UTF-8");

// Define the HTML part.
MimeBodyPart htmlPart = new MimeBodyPart();
htmlPart.setContent(BODY_HTML, "text/html; charset=UTF-8");

// Add the text and HTML parts to the child container.
msg_body.addBodyPart(textPart);
msg_body.addBodyPart(htmlPart);

// Add the child container to the wrapper object.
wrap.setContent(msg_body);

// Create a multipart/mixed parent container.
MimeMultipart msg = new MimeMultipart("mixed");

// Add the parent container to the message.
message.setContent(msg);

// Add the multipart/alternative part to the message.
msg.addBodyPart(wrap);

// Define the attachment
MimeBodyPart att = new MimeBodyPart();
DataSource fds = new FileDataSource(ATTACHMENT);
att.setDataHandler(new DataHandler(fds));
att.setFileName(fds.getName());

// Add the attachment to the message.
msg.addBodyPart(att);

// Try to send the email.
try {
    System.out.println("Attempting to send an email through Amazon SES " +"using the AWS SDK for Java...\n);  
    // Instantiate an Amazon SES client, which will make the service 
    // call with the supplied AWS credentials.
    AmazonSimpleEmailService client = 
        AmazonSimpleEmailServiceClientBuilder.standard() 
        .withRegion(Regions.US_WEST_2).build(); 

    // Print the raw email content on the console
    PrintStream out = System.out;
    message.writeTo(out);

    // Send the email.
    ByteArrayOutputStream outputStream = new ByteArrayOutputStream();
    message.writeTo(outputStream);
    RawMessage rawMessage = 
        new RawMessage(ByteBuffer.wrap(outputStream.toByteArray()));

    SendRawEmailRequest rawEmailRequest = 
        new SendRawEmailRequest(rawMessage) 
        .withConfigurationSetName(CONFIGURATION_SET);
client.sendRawEmail(rawEmailRequest);
System.out.println("Email sent!");
// Display an error if something goes wrong.
} catch (Exception ex) {
    System.out.println("Email Failed");
    System.err.println("Error message: " + ex.getMessage());
    ex.printStackTrace();
}
}

Python

The following code example shows how to use the Python email.mime packages and the AWS SDK for Python (Boto) to compose and send a raw email.

```python
import os
import boto3
from botocore.exceptions import ClientError
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.application import MIMEApplication

# Replace sender@example.com with your "From" address.
# This address must be verified with Amazon SES.
SENDER = "sender Name <sender@example.com>"

# Replace recipient@example.com with a "To" address. If your account
# is still in the sandbox, this address must be verified.
RECIPIENT = "recipient@example.com"

# Specify a configuration set. If you do not want to use a configuration
# set, comment the following variable, and the
# ConfigurationSetName=CONFIGURATION_SET argument below.
CONFIGURATION_SET = "ConfigSet"

# If necessary, replace us-west-2 with the AWS Region you're using for Amazon SES.
AWS_REGION = "us-west-2"

# The subject line for the email.
SUBJECT = "Customer service contact info"

# The full path to the file that will be attached to the email.
ATTACHMENT = "path/to/customers-to-contact.xlsx"

# The email body for recipients with non-HTML email clients.
BODY_TEXT = "Hello,
Please see the attached file for a list of customers to contact."

# The HTML body of the email.
BODY_HTML = """
<html>
<head></head>
<body>
<h1>Hello!</h1>
<p>Please see the attached file for a list of customers to contact.</p>
</body>
</html>
""

# The character encoding for the email.
CHARSET = "utf-8"

# Create a new SES resource and specify a region.
```
client = boto3.client('ses', region_name=AWS_REGION)

# Create a multipart/mixed parent container.
msg = MIMEMultipart('mixed')
# Add subject, from and to lines.
msg['Subject'] = SUBJECT
msg['From'] = SENDER
msg['To'] = RECIPIENT

# Create a multipart/alternative child container.
msg_body = MIMEMultipart('alternative')

# Encode the text and HTML content and set the character encoding. This step is necessary if you're sending a message with characters outside the ASCII range.
textpart = MIMEText(BODY_TEXT.encode(CHARSET), 'plain', CHARSET)
htmlpart = MIMEText(BODY_HTML.encode(CHARSET), 'html', CHARSET)

# Add the text and HTML parts to the child container.
msg_body.attach(textpart)
msg_body.attach(htmlpart)

# Define the attachment part and encode it using MIMEApplication.
att = MIMEApplication(open(ATTACHMENT, 'rb').read())

# Add a header to tell the email client to treat this part as an attachment, and to give the attachment a name.
att.add_header('Content-Disposition', 'attachment', filename=os.path.basename(ATTACHMENT))

# Attach the multipart/alternative child container to the multipart/mixed parent container.
msg.attach(msg_body)

# Add the attachment to the parent container.
msg.attach(att)
# print(msg)

try:
    # Provide the contents of the email.
    response = client.send_raw_email(
        Source=SENDER,
        Destinations=[
            RECIPIENT
        ],
        RawMessage={
            'Data':msg.as_string(),
        },
        ConfigurationSetName=CONFIGURATION_SET
    )

    # Display an error if something goes wrong.
    except ClientError as e:
        print(e.response['Error']['Message'])
    else:
        print("Email sent! Message ID:")
        print(response['MessageId'])

Sending personalized email using the Amazon SES API

You can use the CreateTemplate API operation to create email templates. These templates include a subject line, and the text and HTML parts of the email body. The subject and body sections may also contain unique values that are personalized for each recipient.
There are a few limits and other considerations when using these features:

- You can create up to 10,000 email templates per Amazon SES account.
- Each template can be up to 500KB in size, including both the text and HTML parts.
- You can include an unlimited number of replacement variables in each template.
- You can send email to up to 50 destinations in each call to the `SendBulkTemplatedEmail` operation. A destination includes a list of recipients, as well as CC and BCC recipients. Note that the number of destinations you can contact in a single call to the API may be limited by your account’s maximum sending rate. For more information, see Managing your Amazon SES sending quotas (p. 144).

This section includes procedures for creating email templates and for sending personalized emails.

**Note**

The procedures in this section assume that you’ve already installed and configured the AWS CLI. For more information about installing and configuring the AWS CLI, see the AWS Command Line Interface User Guide.

**Part 1: Set up rendering failure event notifications**

If you send an email that contains invalid personalization content, Amazon SES might accept the message, but won’t be able to deliver it. For this reason, if you plan to send personalized email, you should configure Amazon SES to send Rendering Failure event notifications through Amazon SNS. When you receive a Rendering Failure event notification, you can identify which message contained the invalid content, fix the issues, and send the message again.

The procedure in this section is optional, but highly recommended.

**To configure Rendering Failure event notifications**

1. Create an Amazon SNS topic. For procedures, see Create a Topic in the Amazon Simple Notification Service Developer Guide.
2. Subscribe to the Amazon SNS topic. For example, if you want to receive Rendering Failure notifications by email, subscribe an email endpoint (that is, your email address) to the topic.

   For procedures, see Subscribe to a Topic in the Amazon Simple Notification Service Developer Guide.
3. Complete the procedures in the section called “Set up an Amazon SNS destination” (p. 296) to set up your configuration sets to publish Rendering Failure events to your Amazon SNS topic.

**Part 2: Create an email template**

In this section, you use the CreateTemplate API operation to create a new email template with personalization attributes.

This procedure assumes that you’ve already installed and configured the AWS CLI. For more information about installing and configuring the AWS CLI, see the AWS Command Line Interface User Guide.

**To create the template**

1. In a text editor, create a new file. Paste the following code into the file.

   ```json
   {
     "TemplateName": "MyTemplate",
     "SubjectPart": "Greetings, {{name}}!",
     "EmailPart": "Dear {{name}},
     "Content": "Hello, new email template!
   ```
Sending personalized email

This code contains the following properties:

- **TemplateName** – The name of the template. When you send the email, you refer to this name.
- **SubjectPart** – The subject line of the email. This property may contain replacement tags. These tags use the following format: \{{tagname}\}. When you send the email, you can specify a value for tagname for each destination.

The preceding example includes two tags: {{name}} and {{favoriteanimal}}.
- **HtmlPart** – The HTML body of the email. This property may contain replacement tags.
- **TextPart** – The text body of the email. Recipients whose email clients don't display HTML email see this version of the email. This property may contain replacement tags.

2. Customize the preceding example to fit your needs, and then save the file as mytemplate.json.

3. At the command line, type the following command to create a new template using the CreateTemplate API operation:

```
aws ses create-template --cli-input-json file://mytemplate.json
```

**Part 3: Sending the personalized email**

After you create an email template, you can use it to send email. There are two API operations that you can use to send emails using templates: SendTemplatedEmail, and SendBulkTemplatedEmail.

The SendTemplatedEmail operation is useful for sending a customized email to a single destination (a collection of "To," "CC," and "BCC" recipients who will receive the same email). The SendBulkTemplatedEmail operation is useful for sending unique emails to multiple destinations in a single call to the Amazon SES API. This section provides examples of how to use the AWS CLI to send email using both of these operations.

**Sending templated email to a single destination**

You can use the SendTemplatedEmail operation to send an email to a single destination. All of the recipients in the Destination object will receive the same email.

**To send a templated email to a single destination**

1. In a text editor, create a new file. Paste the following code into the file.

```json
{"Source":"Mary Major <mary.major@example.com>",
 "Template": "MyTemplate",
 "ConfigurationSetName": "ConfigSet",
 "Destination": {
 "ToAddresses": [ "alejandro.rosalez@example.com"
 ],
 "TemplateData": "{ \"name\":\"Alejandro\", \"favoriteanimal\": \"alligator\" }"
}
```

This code contains the following properties:

- **Source** – The email address of the sender.
Sending personalized email

- Template – The name of the template to apply to the email.
- ConfigurationSetName – The name of the configuration set to use when sending the email.

**Note**
We recommend that you use a configuration set that is configured to publish Rendering Failure events to Amazon SNS. For more information, see the section called "Part 1: Set up notifications" (p. 118).

- Destination – The recipient addresses. You can include multiple "To," "CC," and "BCC" addresses. When you use the SendTemplatedEmail operation, all recipients receive the same email.
- TemplateData – An escaped JSON string that contains key-value pairs. The keys correspond to the variables in the template (for example, {{name}}). The values represent the content that replaces the variables in the email.

2. Change the values in the code above to meet your needs, and then save the file as myemail.json.
3. At the command line, type the following command to send the email:

```
aws ses send-templated-email --cli-input-json file://myemail.json
```

Sending templated email to multiple destinations

You can use the SendBulkTemplatedEmail operation to send an email to several destinations in a single call to the API. Amazon SES sends a unique email to the recipient or recipients in each Destination object.

**To send a templated email to multiple destinations**

1. In a text editor, create a new file. Paste the following code into the file.

```json
{
   "Source":"Mary Major <mary.major@example.com>",
   "Template":"MyTemplate",
   "ConfigurationSetName": "ConfigSet",
   "Destinations": [
   {
      "Destination":{
         "ToAddresses": ["anaya.iyengar@example.com"
      },
      "ReplacementTemplateData": "{{name}}: "Anaya", "favoriteanimal": "angelfish"
      },
   {
      "Destination":{
         "ToAddresses": ["liu.jie@example.com"
      },
      "ReplacementTemplateData": "{{name}}: "Liu", "favoriteanimal": "lion"
      },
   {
      "Destination":{
         "ToAddresses": ["shirley.rodriguez@example.com"
      },
      "ReplacementTemplateData": "{{name}}: "Shirley", "favoriteanimal": "shark"
      }
   ]
}
```
This code contains the following properties:

- **Source** – The email address of the sender.
- **Template** – The name of the template to apply to the email.
- **ConfigurationSetName** – The name of the configuration set to use when sending the email.

**Note**

We recommend that you use a configuration set that is configured to publish Rendering Failure events to Amazon SNS. For more information, see the section called “Part 1: Set up notifications” (p. 118).

- **Destinations** – An array that contains one or more Destinations.
- **Destination** – The recipient addresses. You can include multiple "To," "CC," and "BCC" addresses. When you use the SendBulkTemplatedEmail operation, all recipients within the same Destination object receive the same email.
- **ReplacementTemplateData** – A JSON object that contains key-value pairs. The keys correspond to the variables in the template (for example, {{name}}). The values represent the content that replaces the variables in the email.
- **DefaultTemplateData** – A JSON object that contains key-value pairs. The keys correspond to the variables in the template (for example, {{name}}). The values represent the content that replaces the variables in the email. This object contains fallback data. If a Destination object contains an empty JSON object in the ReplacementTemplateData property, the values in the DefaultTemplateData property are used.

2. Change the values in the code above to meet your needs, and then save the file as mybulkemail.json.

3. At the command line, type the following command to send the bulk email:

```
aws ses send-bulk-templated-email --cli-input-json file://mybulkemail.json
```

**Advanced email personalization**

The template feature in Amazon SES is based on the Handlebars template system. You can use Handlebars to create templates that include advanced features, such as nested attributes, array iteration, basic conditional statements, and the creation of inline partials. This section provides examples of these features.

Handlebars includes additional features beyond those documented in this section. For more information, see Built-In Helpers at handlebarsjs.com.

**Topics**

- Parsing nested attributes (p. 122)
- Iterating through lists (p. 122)
- Using basic conditional statements (p. 124)
- Creating inline partials (p. 125)
Parsing nested attributes

Handlebars includes support for nested paths, which makes it easy to organize complex customer data, and then refer to that data in your email templates.

For example, you can organize recipient data into several general categories. Within each of those categories, you can include detailed information. The following code example shows an example of this structure for a single recipient:

```
{
  "meta":{
    "userId":"51806220607"
  },
  "contact":{
    "firstName":"Anaya",
    "lastName":"Iyengar",
    "city":"Bengaluru",
    "country":"India",
    "postalCode":"560052"
  },
  "subscription":[
    {
      "interest":"Sports"
    },
    {
      "interest":"Travel"
    },
    {
      "interest":"Cooking"
    }
  ]
}
```

In your email templates, you can refer to nested attributes by providing the name of the parent attribute, followed by a period (.), followed by the name of the attribute for which you want to include the value. For example, if you use the data structure shown in the preceding example, and you want to include each recipient's first name in the email template, include the following text in your email template:

```
Hello {{contact.firstName}}!
```

Handlebars can parse paths that are nested several levels deep, which means you have flexibility in how you structure your template data.

Iterating through lists

The `each` helper function iterates through items in an array. The following code is an example of an email template that uses the `each` helper function to create an itemized list of each recipient's interests.

```
{
  "Template": {
    "TemplateName": "Preferences",
    "SubjectPart": "Subscription Preferences for {{contact.firstName}} {{contact.lastName}}",
    "HtmlPart": "<h1>Your Preferences</h1>
    <p>You have indicated that you are interested in receiving information about the following subjects:</p>
    <ul>
      {{#each subscription}}
      <li>{{interest}}</li>
      {{/each}}
    </ul>
    <p>You can change these settings at any time by visiting

```

122
Sending personalized email

Important
In the preceding code example, the values of the HtmlPart and TextPart attributes include line breaks to make the example easier to read. The JSON file for your template can't contain line breaks within these values. If you copied and pasted this example into your own JSON file, remove the line breaks and extra spaces from the HtmlPart and TextPart sections before proceeding.

After you create the template, you can use the SendTemplatedEmail or the SendBulkTemplatedEmail operation to send email to recipients using this template. As long as each recipient has at least one value in the Interests object, they receive an email that includes an itemized list of their interests. The following example shows a JSON file that can be used to send email to multiple recipients using the preceding template:

```
{
  "Source":"Sender Name <sender@example.com>",
  "Template":"Preferences",
  "Destinations":[
    {
      "Destination":{
        "ToAddresses":[
          "anaya.iyengar@example.com"
        ]
      },
      "ReplacementTemplateData":"{"meta":{"userId":"51806220607"},"contact":{"firstName":"Anaya","lastName":"Iyengar"},"subscription":[{"interest":"Sports"},{"interest":"Travel"},{"interest":"Cooking"}]}
    },
    {
      "Destination":{
        "ToAddresses":[
          "shirley.rodriguez@example.com"
        ]
      },
      "ReplacementTemplateData":"{"meta":{"userId":"1981624758263"},"contact":{"firstName":"Shirley","lastName":"Rodriguez"},"subscription":[{"interest":"Technology"},{"interest":"Politics"}]}
    }
  ],
  "DefaultTemplateData":{"meta":{"userId":""},"contact":{"firstName":"Friend","lastName":""},"subscription":[[]]}
}
```

When you send an email to the recipients listed in the preceding example using the SendBulkTemplatedEmail operation, they receive a message that resembles the example shown in the following image:
Using basic conditional statements

This section builds on the example described in the previous section. The example in the previous section uses the `each` helper to iterate through a list of interests. However, recipients for whom no interests are specified receive an email that contains an empty list. By using the `{{if}}` helper, you can format the email differently if a certain attribute is present in the template data. The following code uses the `{{if}}` helper to display the bulleted list from the preceding section if the Subscription array contains any values. If the array is empty, a different block of text is displayed.

```json
{
    "TemplateName": "Preferences2",
    "SubjectPart": "Subscription Preferences for {{contact.firstName}} {{contact.lastName}}",
    "HtmlPart": "<h1>Your Preferences</h1>
    <p>Dear {{contact.firstName}},</p>
    {{#if subscription}}
    <p>You have indicated that you are interested in receiving information about the following subjects:</p>
    <ul>
    {{#each subscription}}
    <li>{{interest}}</li>
    {{/each}}
    </ul>
    <p>You can change these settings at any time by visiting the <a href="https://www.example.com/preferences/i.aspx?id={{meta.userId}}">Preference Center</a>.</p>
    {{else}}
    <p>Please update your subscription preferences by visiting the <a href="https://www.example.com/preferences/i.aspx?id={{meta.userId}}">Preference Center</a>.</p>
    {{/if}}",
    "TextPart": "Your Preferences

Dear {{contact.firstName}},

{{#if subscription}}
You have indicated that you are interested in receiving information about the following subjects:

- {{interest}}

You can change these settings at any time by visiting the Preference Center at https://www.example.com/preferences/i.aspx?id={{meta.userId}}.
{{else}}
Please update your subscription preferences by visiting the Preference Center at https://www.example.com/preferences/i.aspx?id={{meta.userId}}.
{{/if}}
"}
```
Important

In the preceding code example, the values of the HtmlPart and TextPart attributes include line breaks to make the example easier to read. The JSON file for your template can't contain line breaks within these values. If you copied and pasted this example into your own JSON file, remove the line breaks and extra spaces from the HtmlPart and TextPart sections before proceeding.

The following example shows a JSON file that can be used to send email to multiple recipients using the preceding template:

```json
{
   "Source":"Sender Name <sender@example.com>",
   "Template":"Preferences2",
   "Destinations":[
      {
         "Destination":{
            "ToAddresses":[
               "anaya.iyengar@example.com"
            ],
            "ReplacementTemplateData":"{"meta":{"userId":"51806220607"},"contact":{"firstName":"Anaya","lastName":"Iyengar"},"subscription":[{"interest":"Sports"},{"interest":"Cooking"}]}",
            "DefaultTemplateData":null
         }
      },
      {
         "Destination":{
            "ToAddresses":[
               "shirley.rodriguez@example.com"
            ],
            "ReplacementTemplateData":"{"meta":{"userId":"1981624758263"},"contact":{"firstName":"Shirley","lastName":"Rodriguez"}}",
            "DefaultTemplateData":null
         }
      }
   ],
   "DefaultTemplateData":null
}
```

In this example, the recipient whose template data included a list of interests receives the same email as the example shown in the previous section. The recipient whose template data did not include any interests, however, receives an email that resembles the example shown in the following image:

![Image of email example]

Creating inline partials

You can use inline partials to simplify templates that include repeated strings. For example, you could create an inline partial that includes the recipient's first name, and, if it is available, their last name by adding the following code to the beginning of your template:

```html
{{#* inline "fullName"}}{{firstName}}{{#if lastName}} {{lastName}}{{/if}}{{/inline}}
```

Note

The newline character (\n) is required to separate the {{inline}} block from the content in your template. The newline isn't rendered in the final output.
After you create the `fullName` partial, you can include it anywhere in your template by preceding the name of the partial with a greater-than (> ) sign followed by a space, as in the following example: `{{> fullName}}`. Inline partials are not transferred between parts of the email. For example, if you want to use the same inline partial in both the HTML and the text version of the email, you must define it in both the `HtmlPart` and the `TextPart` sections.

You can also use inline partials when iterating through arrays. You can use the following code to create a template that uses the `fullName` inline partial. In this example, the inline partial applies to both the recipient's name and to an array of other names:

```json
{
  "TemplateName": "Preferences3",
  "SubjectPart": "{{firstName}}'s Subscription Preferences",
  "HtmlPart": "{{#* inline "fullName"}}
    {{firstName}}{{#if lastName}} {{lastName}}{{/if}}
    {{/inline~}}
    <h1>Hello {{> fullName}}!</h1>
    <p>You have listed the following people as your friends:</p>
    <ul>
    {{#each friends}}
      <li>{{> fullName}}</li>
    {{/each}}</ul>
  {{/inline-}}
}
```

**Important**

In the preceding code example, the values of the `HtmlPart` and `TextPart` attributes include line breaks to make the example easier to read. The JSON file for your template can't contain line breaks within these values. If you copied and pasted this example into your own JSON file, remove the line breaks and extra spaces from these sections.

### Managing email templates

In addition to creating email templates (p. 117), you can also use the Amazon SES API to update or delete existing templates, to list all of your existing templates, or to view the contents of a template.

This section contains procedures for using the AWS CLI to perform tasks related to Amazon SES templates.

**Note**

The procedures in this section assume that you've already installed and configured the AWS CLI. For more information about installing and configuring the AWS CLI, see the AWS Command Line Interface User Guide.

### Viewing a list of email templates

You can use the `ListTemplates` operation in the Amazon SES API to view a list of all of your existing email templates.

**To view a list of email templates**

- At the command line, enter the following command:
Sending personalized email

```
aws ses list-templates
```

If there are existing email templates in your Amazon SES account in the current Region, this command returns a response that resembles the following example:

```
{
   "TemplatesMetadata": [
      {
         "Name": "SpecialOffers",
         "CreatedTimestamp": "2020-08-05T16:04:12.640Z"
      },
      {
         "Name": "NewsAndUpdates",
         "CreatedTimestamp": "2019-10-03T20:03:34.574Z"
      }
   ]
}
```

If you haven't created any templates, the command returns a `TemplatesMetadata` object with no members.

**Viewing the contents of a specific email template**

You can use the `GetTemplate` operation in the Amazon SES API to view the contents of a specific email template.

**To view the contents of an email template**

- At the command line, enter the following command:

```
aws ses get-template --template-name MyTemplate
```

In the preceding command, replace `MyTemplate` with the name of the template that you want to view.

If the template name that you provided matches a template that exists in your Amazon SES account, this command returns a response that resembles the following example:

```
{
   "Template": {
      "TemplateName": "TestMessage",
      "SubjectPart": "Amazon SES Test Message",
      "TextPart": "Hello! This is the text part of the message."
      "HtmlPart": "<html><body><h2>Hello!</h2><p>This is the HTML part of the message.</p></body></html>"
   }
}
```

If the template name that you provided doesn't match a template that exists in your Amazon SES account, the command returns a TemplateDoesNotExist error.

**Deleting an email template**

You can use the `DeleteTemplate` operation in the Amazon SES API to delete a specific email template.
To delete an email template

- At the command line, enter the following command:

```bash
aws ses delete-template --template-name MyTemplate
```

In the preceding command, replace `MyTemplate` with the name of the template that you want to delete.

This command doesn't provide any output. You can verify that the template was deleted by using the `GetTemplate (p. 127)` operation.

Updating an email template

You can use the `UpdateTemplate` operation in the Amazon SES API to update an existing email template. For example, this operation is helpful if you want to change the subject line of the email template, or if you need to modify the body of the message itself.

To update an email template

1. Use the `GetTemplate` command to retrieve the existing template by entering the following command on the command line:

```bash
aws ses get-template --template-name MyTemplate
```

In the preceding command, replace `MyTemplate` with the name of the template that you want to update.

If the template name that you provided matches a template that exists in your Amazon SES account, this command returns a response that resembles the following example:

```json
{
  "Template": {
    "TemplateName": "TestMessage",
    "SubjectPart": "Amazon SES Test Message",
    "TextPart": "Hello! This is the text part of the message."
  }
}
```

2. In a text editor, create a new file. Paste the output of the previous command into the file.

3. Modify the template as needed. Any lines that you omit are removed from the template. For example, if you only want to change the `SubjectPart` of the template, you still need to include the `TextPart` and `HtmlPart` properties.

   When you finish, save the file as `update_template.json`.

4. At the command line, enter the following command:

```bash
aws ses update-template --cli-input-json file://path/to/update_template.json
```

In the preceding command, replace `path/to/update_template.json` with the path to the `update_template.json` file that you created in the previous step.

If the template is updated successfully, this command doesn't provide any output. You can verify that the template was updated by using the `GetTemplate (p. 127)` operation.
Amazon Simple Email Service Classic Developer Guide

Email authentication methods

If the template that you specified doesn’t exist, this command returns a TemplateDoesNotExist error. If the template doesn’t contain either the TextPart or HtmlPart property (or both), this command returns an InvalidParameterValue error.

Authenticating your email in Amazon SES

Amazon Simple Email Service (Amazon SES) uses the Simple Mail Transfer Protocol (SMTP) to send email. Because SMTP does not provide any authentication by itself, spammers can send email messages that claim to originate from someone else, while hiding their true origin. By falsifying email headers and spoofing source IP addresses, spammers can mislead recipients into believing that the email messages that they are receiving are authentic.

Most ISPs that forward email traffic take measures to evaluate whether email is legitimate. One such measure that ISPs take is to determine whether an email is authenticated. Authentication requires senders to verify that they are the owner of the account that they are sending from. In some cases, ISPs refuse to forward email that is not authenticated. To ensure optimal deliverability, we recommend that you authenticate your emails.

The following sections describe two authentication mechanisms ISPs use—Sender Policy Framework (SPF) and DomainKeys Identified Mail (DKIM)—and provide instructions for how to use these standards with Amazon SES.

• To learn about SPF, which provides a way to trace an email message back to the system from which it was sent, see Authenticating Email with SPF in Amazon SES (p. 129).
• To learn about DKIM, a standard that allows you to sign your email messages to show ISPs that your messages are legitimate and have not been modified in transit, see Authenticating Email with DKIM in Amazon SES (p. 130).
• To learn how to comply with Domain-based Message Authentication, Reporting and Conformance (DMARC), which relies on SPF and DKIM, see Complying with DMARC Using Amazon SES (p. 142).

Authenticating Email with SPF in Amazon SES

Sender Policy Framework (SPF) is an email validation standard that’s designed to prevent email spoofing. Domain owners use SPF to tell email providers which servers are allowed to send email from their domains. SPF is defined in RFC 7208.

To set up SPF, you publish a TXT record to the DNS configuration for your domain. This record contains a list of the servers that you authorize to send email from your domain. When an email provider receives a message from your domain, it checks the DNS records for your domain to make sure that the email was sent from an authorized server.

When you send email through Amazon SES, the messages that you send pass an SPF check by default. Amazon SES specifies a MAIL FROM domain for each message that is a subdomain of amazonses.com, and the sending mail server for the message aligns with this domain.

You can optionally publish your own SPF record. By publishing an SPF record, your email can comply with Domain-based Message Authentication, Reporting and Conformance (DMARC). For more information, see Complying with DMARC (p. 142).

Adding an SPF Record

To publish an SPF record, you have to add a new TXT record to the DNS configuration for your domain. The procedures for updating DNS records vary depending on which DNS or web hosting provider you use.
The following table includes links to the documentation for several common providers. This list isn’t exhaustive, and inclusion in this list isn’t an endorsement or recommendation of any company’s products or services. If your provider isn’t listed in the table, you can probably still publish an SPF record.

<table>
<thead>
<tr>
<th>DNS/Hosting Provider</th>
<th>Documentation Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Route 53</td>
<td>Creating Records by Using the Amazon Route 53 Console</td>
</tr>
<tr>
<td>GoDaddy</td>
<td>Add an SPF record</td>
</tr>
<tr>
<td>Dreamhost</td>
<td>How do I add an SPF record?</td>
</tr>
<tr>
<td>Cloudflare</td>
<td>Managing DNS records in CloudFlare</td>
</tr>
<tr>
<td>HostGator</td>
<td>SPF Records</td>
</tr>
<tr>
<td>Namecheap</td>
<td>How do I add TXT/SPF/DKIM/DMARC records for my domain?</td>
</tr>
<tr>
<td>Names.co.uk</td>
<td>Changing your domain's DNS settings</td>
</tr>
<tr>
<td>Wix</td>
<td>Adding or Updating SPF Records in Your Wix Account</td>
</tr>
</tbody>
</table>

If your domain doesn't have an existing SPF record, publish a TXT record with the following value. The name of the record can be blank or @, depending on your DNS service.

```
"v=spf1 include:amazonses.com -all"
```

SPF records can contain multiple `include` statements. If your domain already has an SPF record, you can add an `include` statement for Amazon SES by using the following format:

```
"v=spf1 include:example.com include:amazonses.com -all"
```

**Authenticating Email with DKIM in Amazon SES**

*DomainKeys Identified Mail (DKIM)* is a standard that allows senders to sign their email messages with a cryptographic key. Email providers then use these signatures to verify that the messages weren't modified by a third party while in transit.

An email message that is sent using DKIM includes a *DKIM-Signature* header field that contains a cryptographically signed representation of the message. A provider that receives the message can use a public key, which is published in the sender's DNS record, to decode the signature. Email providers then use this information to determine whether messages are authentic.

DKIM signatures are optional. You might decide to sign your email using a DKIM signature to enhance deliverability with DKIM-compliant email providers. Amazon SES provides three options for signing your messages using a DKIM signature:

- To set up a sending identity (such as a domain or an email address) so that Amazon SES automatically adds a DKIM signature to every message that you send from that identity, see *Easy DKIM in Amazon SES* (p. 131).
- To use your own public-private key pair for DKIM authentication, see *Provide Your Own DKIM Authentication Token in Amazon SES* (p. 138).
To add your own DKIM signature to email that you send using the SendRawEmail API, see Manual DKIM Signing in Amazon SES (p. 141).

Easy DKIM in Amazon SES

When you set up Easy DKIM for an identity, Amazon SES automatically adds a 1024-bit DKIM key to every email that you send from that identity. You can configure Easy DKIM by using the Amazon SES console, or by using the API.

Note
To set up Easy DKIM, you have to modify the DNS settings for your domain. If you use Route 53 as your DNS provider, Amazon SES can automatically create the appropriate records for you. If you use another DNS provider, see your provider's documentation to learn more about changing the DNS settings for your domain.

When you successfully configure Easy DKIM, you can start sending email from the DKIM enabled domain, even if you haven't completed the procedures in Verifying a domain with Amazon SES (p. 60).

Easy DKIM Considerations

When you use Easy DKIM to authenticate your email, the following rules apply:

- You only need to set up Easy DKIM for the domain that you use in your "From" address. You don't need to set up Easy DKIM for domains that you use in "Return-Path" or "Reply-to" addresses.
- Amazon SES is available in several AWS Regions. If you use more than one AWS Region to send email, you have to complete the Easy DKIM setup process in each of those regions to ensure that all of your email is DKIM-signed.
- When you verify a domain, your Easy DKIM settings also apply to all subdomains of that domain, unless you set up Easy DKIM for specific subdomains.
- If you set up Easy DKIM for a parent domain, a subdomain, and an email address, Amazon SES applies Easy DKIM settings in the following ways:
  - DKIM settings for a subdomain override the settings for the parent domain.
  - DKIM settings for an email address override the settings for the subdomain (if applicable) and the parent domain.

This topic contains the following sections:
- Setting Up Easy DKIM for a Domain (p. 131)
- Setting Up Easy DKIM for an Email Address (p. 133)
- Managing Easy DKIM (p. 134)

Setting Up Easy DKIM for a Domain

The procedure in this section shows you how to set up Easy DKIM for a domain. If you setup Easy DKIM for a domain, then you can start sending email from that domain, even if you haven't completed the procedure to verify a domain (p. 59).

To set up Easy DKIM for a domain

1. Open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, under Identity Management, choose Domains.
3. In the list of domains, choose the domain that you want to set up Easy DKIM for.

Note
If you haven't started the verification process for the domain yet, see the procedures at Verifying a domain with Amazon SES (p. 60).
4. Under **DKIM**, choose **Generate DKIM Settings**.

5. Copy the three CNAME records that appear in this section. Alternatively, you can choose **Download Record Set as CSV** to save a copy of the records to your computer.

The following image shows an example of the **DKIM** section.

6. Add the CNAME records to the DNS configuration for your domain. To update the DNS records for your domain:

   - **If you use Route 53 as your DNS provider** – If you use Route 53 on the same account that you use when you send email using Amazon SES, choose **Use Route 53** to automatically update the DNS settings for your domain. Otherwise, complete the procedures shown in **Editing Records** in the Amazon Route 53 Developer Guide.

   - **If you use another DNS provider** – Different providers have different procedures for updating DNS records. The following table lists links to the documentation for several common providers. This list isn't exhaustive and inclusion in this list isn't an endorsement or recommendation of any company's products or services. If your provider isn't listed in the table, you can probably use the domain with Amazon SES.

<table>
<thead>
<tr>
<th>DNS/Hosting Provider</th>
<th>Documentation Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>GoDaddy</td>
<td>Add a CNAME record</td>
</tr>
</tbody>
</table>
Setting Up Easy DKIM for an Email Address

The procedure in this section shows how to set up Easy DKIM for a specific email address that you've already verified with Amazon SES. You can only configure Easy DKIM for email addresses that belong to domains you already own, because you have to change the DNS settings for the domain in order to set up Easy DKIM for an email address.

**Important**
You can't set up Easy DKIM for email addresses on domains that you don't own. For example, you can't set up Easy DKIM for a gmail.com or hotmail.com address.

If you already set up Easy DKIM for the domain that the email address belongs to, you don't need to set up Easy DKIM for the email address as well. When you set up Easy DKIM for a domain, Amazon SES automatically authenticates every email from every address on that domain. Easy DKIM settings for a specific email address automatically override the settings for the domain that the address belongs to.

To set up Easy DKIM for an email address

1. Open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, under **Identity Management**, choose **Email Addresses**.
3. In the list of email addresses, choose the address that you want to set up Easy DKIM for.
4. Under **DKIM**, choose **Generate DKIM Settings**.
5. Copy the three CNAME records that appear in this section. Alternatively, you can choose **Download Record Set as CSV** to save a copy of the records to your computer.
6. Add the CNAME records to the DNS configuration for your domain. To update the DNS records for your domain:
   - If you use Route 53 as your DNS provider – Complete the procedures shown in **Editing Records** in the Amazon Route 53 Developer Guide.
• If you use another DNS provider – Different providers have different procedures for updating DNS records. See the documentation provided by your DNS provider for more information.

  **Note**
  A small number of DNS providers don't allow you to include underscores (_) in record names. However, the underscore in the DKIM record name is required. If your DNS provider doesn't allow you to enter an underscore in the record name, contact the provider's customer support team for assistance.

• If you're not sure who your DNS provider is – Ask your system administrator for more information.

Amazon SES usually detects changes to your DNS configuration within 72 hours.

**Managing Easy DKIM**

There are two ways to manage the Easy DKIM settings for your identities: by using the web-based Amazon SES console, or by using the Amazon SES API. You can use either of these methods to obtain the DKIM records for an identity, or to enable or disable Easy DKIM for an identity.

**Obtaining Easy DKIM Records for An Identity**

You can obtain the Easy DKIM records for your domain or email address at any time by using the Amazon SES console.

**To obtain the Easy DKIM records for an identity by using the console**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, under **Identity Management**, choose the type of identity that you want to obtain Easy DKIM records for.
3. In the list of identities, choose the identity that you want to obtain Easy DKIM records for.
4. In the **DKIM** section, copy the three CNAME records. The following image shows an example of the **DKIM** section.
You can also obtain the CNAME records for an identity by using the Amazon SES API. A common method of interacting with the API is to use the AWS CLI.

To obtain the Easy DKIM records for an identity by using the AWS CLI

1. At the command line, type the following command:

   ```bash
   aws ses get-identity-dkim-attributes --identities "example.com"
   ```

   In the preceding example, replace `example.com` with the identity that you want to obtain Easy DKIM records for. You can specify either an email address or a domain.

2. The output of this command contains a `DkimTokens` section, as shown in the following example:

   ```json
   {
     "DkimAttributes": {
       "example.com": {
         "DkimEnabled": true,
         "DkimVerificationStatus": "Success",
         "DkimTokens": [
           "hirjd4exampled5477y22yd23ettobiho_.domainkey.example.com",
           "v3rnz522czcl46queexamplek3efo506x_.domainkey.example.com",
           "y4examplexbhyhnsjcmtvzotfvqjmdqoj_.domainkey.example.com"
         ]
       }
     }
   }
   ```
You can use the tokens to create the CNAME records that you add to the DNS settings for your domain. To create the CNAME records, use the following template:

```
<token1>.domainkey.example.com CNAME <token1>.dkim.amazonses.com
<token2>.domainkey.example.com CNAME <token2>.dkim.amazonses.com
<token3>.domainkey.example.com CNAME <token3>.dkim.amazonses.com
```

Replace each instance of `token1` with the first token in the list you received when you ran the `aws ses get-identity-dkim-attributes` command, replace all instances of `token2` with the second token in the list, and replace all instances of `token3` with the third token in the list.

For example, applying this template to the tokens shown in the preceding example produces the following records:

```
hirjd4exampled5477y22yd23ettobi._domainkey.example.com CNAME hirjd4exampled5477y22yd23ettobi.dkim.amazonses.com
v3rnz522czcl46quexamplek3efo5o6x._domainkey.example.com CNAME v3rnz522czcl46quexamplek3efo5o6x.dkim.amazonses.com
y4examplexbhyhnsjcmtvzotfvqjmdqoj._domainkey.example.com CNAME y4examplexbhyhnsjcmtvzotfvqjmdqoj.dkim.amazonses.com
```

### Disabling Easy DKIM for an Identity

You can quickly disable DKIM authentication for an identity by using the Amazon SES console.

#### To disable Easy DKIM for an identity

1. Sign in to the AWS Management Console and open the Amazon SES console at `https://console.aws.amazon.com/ses/`.
2. In the navigation pane, under **Identity Management**, choose the type of identity that you want to disable Easy DKIM for.
3. In the list of identities, choose the identity that you want to disable Easy DKIM for.
4. In the **DKIM** section, next to **DKIM: enabled**, choose **disable**, as shown in the following image.
You can also disable Easy DKIM for an identity by using the Amazon SES API. A common method of interacting with the API is to use the AWS CLI.

**To disable Easy DKIM for an identity by using the AWS CLI**

- At the command line, type the following command:

```
aws ses set-identity-dkim-enabled --identity example.com --no-dkim-enabled
```

In the preceding example, replace `example.com` with the identity that you want to disable Easy DKIM for. You can specify either an email address or a domain.

**Enabling Easy DKIM for an Identity**

If you previously disabled Easy DKIM for an identity, you can enable it again by using the Amazon SES console.

**To enable Easy DKIM for an identity**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, under **Identity Management**, choose the type of identity that you want to enable Easy DKIM for.
3. In the list of identities, choose the identity that you want to enable Easy DKIM for.
4. In the **DKIM** section, next to **DKIM: disabled**, choose **enable**, as shown in the following image.

You can also enable Easy DKIM for an identity by using the Amazon SES API. A common method of interacting with the API is to use the AWS CLI.

**To enable Easy DKIM for an identity by using the AWS CLI**

- At the command line, type the following command:

```
aws ses set-identity-dkim-enabled --identity example.com --dkim-enabled
```

In the preceding example, replace `example.com` with the identity that you want to enable Easy DKIM for. You can specify either an email address or a domain.
Provide Your Own DKIM Authentication Token in Amazon SES

As an alternative to using Easy DKIM (p. 131), you can instead configure DKIM authentication by using your own public-private key pair. This process is known as Bring Your Own DKIM (BYODKIM).

With BYODKIM, you can use a single DNS record to configure DKIM authentication for your domains, as opposed to Easy DKIM, which requires you to publish three separate DNS records. Additionally, using BYODKIM lets you rotate the DKIM keys for your domains as often as you want.

Topics in this section:
- Step 1: Create the Key Pair (p. 138)
- Step 2: Add the Public Key to the DNS Configuration for Your Domain (p. 138)
- Step 3: Configure a Domain to Use BYODKIM (p. 139)

**Step 1: Create the Key Pair**

To use the Bring Your Own DKIM feature, you first have to create a key pair.

The private key that you generate has to use 1024-bit RSA encoding. The private key has to be in PKCS #1 format.

This section shows you how to use the `openssl` command that's built in to most Linux, macOS, or Unix operating systems to create the key pair.

**Note**

If you use a Windows computer, you can use third-party applications to generate RSA key pairs. If you use a third-party application, it has to be able to generate a 1024-bit RSA key pair in PKCS #1 format.

To create the key pair from the Linux, macOS, or Unix command line

1. At the command line, enter the following command to generate the private key:

   ```bash
   openssl genrsa -f4 -out private.key 1024
   ```

2. At the command line, enter the following command to generate the public key:

   ```bash
   openssl rsa -in private.key -outform PEM -pubout -out public.key
   ```

**Step 2: Add the Public Key to the DNS Configuration for Your Domain**

Now that you've created a key pair, you have to add the public key to the DNS configuration for your domain as a TXT record.

To add the public key to the DNS configuration for your domain

1. Sign in to the management console for your DNS or hosting provider.
2. Add a new text record to the DNS configuration for your domain. The record should use the following format:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>selector._domainkey.example.com</td>
<td>TXT</td>
<td>p=yourPublicKey</td>
</tr>
</tbody>
</table>

138
In the preceding example, make the following changes:

- Replace `selector` with a unique name that identifies the key.
- Replace `example.com` with your domain.
- Replace `yourPublicKey` with the public key that you created earlier.

**Note**
You have to delete the first and last lines (`-----BEGIN PUBLIC KEY-----` and `-----END PUBLIC KEY-----`, respectively) of the generated public key. Additionally, you have to remove the line breaks in the generated public key. The resulting value is a string of characters with no spaces or line breaks.

Different providers have different procedures for updating DNS records. The following table lists links to the documentation for several common providers. This list isn’t exhaustive and inclusion in this list isn’t an endorsement or recommendation of any company’s products or services.

<table>
<thead>
<tr>
<th>DNS/Hosting Provider</th>
<th>Documentation Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Route 53</td>
<td>Editing Records in the Amazon Route 53 Developer Guide</td>
</tr>
<tr>
<td>GoDaddy</td>
<td>Add a TXT record</td>
</tr>
<tr>
<td>Dreamhost</td>
<td>How do I add custom DNS records?</td>
</tr>
<tr>
<td>Cloudflare</td>
<td>Managing DNS records in CloudFlare</td>
</tr>
<tr>
<td>HostGator</td>
<td>Manage DNS Records with HostGator/eNom</td>
</tr>
<tr>
<td>Namecheap</td>
<td>How do I add TXT/SPF/DKIM/DMARC records for my domain?</td>
</tr>
<tr>
<td>Names.co.uk</td>
<td>Changing your domain's DNS settings</td>
</tr>
<tr>
<td>Wix</td>
<td>Adding or Updating TXT Records in Your Wix Account</td>
</tr>
</tbody>
</table>

**Step 3: Configure a Domain to Use BYODKIM**

You can set up BYODKIM for both new domains (that is, domains that you don't currently use to send email through Amazon SES) and existing domains (that is, domains that you’ve already set up to use with Amazon SES). To set up a new domain, use the `CreateEmailIdentity` operation in the Amazon SES API. To configure an existing domain, use the `PutEmailIdentityDkimSigningAttributes` operation.

This section includes procedures for setting up new and existing domains by using the AWS CLI. Before you complete the procedures in this section, you first have to install and configure the AWS CLI. For more information, see the [AWS Command Line Interface User Guide](https://docs.aws.amazon.com/cli/latest/userguide/).  

**Option 1: Creating a New Domain Identity That Uses BYODKIM**

This section contains a procedure for creating a new domain identity that uses BYODKIM. A new domain identity is a domain that you haven't previously set up to send email using Amazon SES.

If you want to configure an existing domain to use BYODKIM, complete the procedure in **Option 2: Configuring an Existing Domain Identity (p. 140)** instead.
To create the identity

1. In a text editor, paste the following code:

```json
{
  "EmailIdentity": "example.com",
  "DkimSigningAttributes": {
    "DomainSigningPrivateKey": "privateKey",
    "DomainSigningSelector": "selector"
  }
}
```

In the preceding example, make the following changes:

- Replace `example.com` with the domain that you want to create.
- Replace `privateKey` with your private key.
- Replace `selector` with the unique selector that you specified when you created the TXT record in the DNS configuration for your domain.

When you finish, save the file as `create-identity.json`.

2. At the command line, enter the following command:

```bash
aws sesv2 create-email-identity --cli-input-json file://path/to/create-identity.json
```

In the preceding command, replace `path/to/create-identity.json` with the complete path to the file that you created in the previous step.

Option 2: Configuring an Existing Domain Identity

This section contains a procedure for updating an existing domain identity to use BYODKIM. A an existing domain identity is a domain that you have already set up to send email using Amazon SES.

To update the domain identity

1. In a text editor, paste the following code:

```json
{
  "SigningAttributes": {
    "DomainSigningPrivateKey": "privateKey",
    "DomainSigningSelector": "selector"
  },
  "SigningAttributesOrigin": "EXTERNAL"
}
```

In the preceding example, make the following changes:

- Replace `privateKey` with your private key.
- Replace `selector` with the unique selector that you specified when you created the TXT record in the DNS configuration for your domain.

When you finish, save the file as `update-identity.json`.

2. At the command line, enter the following command:
Authenticating Email with DKIM

In the preceding command, make the following changes:

- Replace `path/to/update-identity.json` with the complete path to the file that you created in the previous step.
- Replace `example.com` with the domain that you want to update.

Checking the DKIM Status for a Domain That Uses BYODKIM

After you configure a domain to use BYODKIM, you can use the GetEmailIdentity operation to confirm that DKIM is properly configured.

To check the DKIM status of a domain

- At the command line, enter the following command:

```
aws sesv2 get-email-identity --email-identity example.com
```

In the preceding command, replace `example.com` with your domain.

This command returns a JSON object that contains a section that resembles the following example.

```
{
  ...
  "DkimAttributes": {
    "SigningAttributesOrigin": "EXTERNAL",
    "SigningEnabled": true,
    "Status": "SUCCESS",
    "Tokens": [ ]
  },
  ...
}
```

BYODKIM is properly configured for the domain if all of the following are true:

- The value of the `SigningAttributesOrigin` property is `EXTERNAL`.
- The value of `SigningEnabled` is `true`.
- The value of `Status` is `SUCCESS`.

Manual DKIM Signing in Amazon SES

As an alternative to using Easy DKIM, you can instead manually add DKIM signatures to your messages, and then send those messages using Amazon SES. If you choose to manually sign your messages, you first have to create a DKIM signature. After you create the message and the DKIM signature, you can use the SendRawEmail API to send it.

If you decide to manually sign your email, consider the following factors:

- Every message that you send by using Amazon SES contains a DKIM header that references a signing domain of `amazonses.com` (that is, it contains the following string: `d=amazonses.com`). If you manually sign your messages, your messages should include two DKIM headers: one for your domain, and one for `amazonses.com`.
• Amazon SES doesn't validate DKIM signatures that you manually add to your messages. If there are errors with the DKIM signature in a message, it might be rejected by email providers.
• When you sign your messages, you should use a bit length of at least 1024 bits.
• Don't sign the following fields: Message-ID, Date, Return-Path, Bounces-To.

Note
If you use an email client to send email using the Amazon SES SMTP interface, your client might automatically perform DKIM signing of your messages. Some clients might sign some of these fields. See the documentation for your email client to see which fields are signed by default.

Complying with DMARC Using Amazon SES

Domain-based Message Authentication, Reporting and Conformance (DMARC) is an email authentication protocol that uses Sender Policy Framework (SPF) and DomainKeys Identified Mail (DKIM) to detect email spoofing. In order to comply with DMARC, messages must be authenticated through either SPF or DKIM, or both.

This topic contains information that will help you configure Amazon SES so that the emails you send comply with both SPF and DKIM. By complying with one of these authentication systems, your emails will comply with DMARC. For information about the DMARC specification, see http://www.dmarc.org.

Setting Up the DMARC Policy on Your Domain

To set up DMARC, you have to modify the DNS settings for your domain. The DNS settings for your domain should include a TXT record that specifies the domain's DMARC settings. The procedures for adding TXT records to your DNS configuration depend on which DNS or hosting provider you use. If you use Route 53, see Working with Records in the Amazon Route 53 Developer Guide. If you use another provider, see the DNS configuration documentation for your provider.

The name of the TXT record you create should be _dmarc.example.com, where example.com is your domain. The value of the TXT record contains the DMARC policy that applies to your domain. The following is an example of a TXT record that contains a DMARC policy:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>_dmarc.example.com</td>
<td>TXT</td>
<td>&quot;v=DMARC1;p=quarantine;pct=25;rua=<a href="mailto:dmarcreports@example.com">mailto:dmarcreports@example.com</a>&quot;</td>
</tr>
</tbody>
</table>

In plain language, this policy tells email providers to do the following:

• Look for all emails with a "From" domain of example.com that don't pass SPF or DKIM authentication.
• Quarantine 25% of the emails that failed authentication by sending them to the Spam folder (you can also do nothing by using p=none, or reject the messages outright by using p=reject).
• Send reports about all emails that failed authentication in a digest (that is, a report that aggregates the data for a certain time period, rather than sending individual reports for each event). Email providers typically send these aggregated reports once per day, although these policies differ from provider to provider.

To learn more about configuring DMARC for your domain, see the Overview on the DMARC website.

For complete specifications of the DMARC system, see RFC 7489 on the IETF website. Section 6.3 of this document contains a complete list of tags that you can use to configure the DMARC policy for your domain.
Complying with DMARC through SPF

For an email to comply with DMARC based on SPF, both of the following conditions must be met:

- The email must pass an SPF check.
- The domain in the From address of the email header must align with the MAIL FROM domain that the sending mail server specifies to the receiving mail server. If the domain's DMARC policy for SPF specifies strict alignment, the From and MAIL FROM domains must match exactly. If the domain's DMARC policy for SPF specifies relaxed alignment, the MAIL FROM domain can be a subdomain of the domain in the From header.

To comply with these requirements, complete the following steps:

- Set up a custom MAIL FROM domain by completing the procedures in the section called “Setting up a custom MAIL FROM domain” (p. 65).
- Ensure that your sending domain uses a relaxed policy for SPF. If you have not changed your domain's policy alignment, it will use a relaxed policy by default.

  Note
  You can determine your domain's DMARC alignment for SPF by typing the following command at the command line, replacing `example.com` with your domain:

  ```
  nslookup -type=TXT _dmarc.example.com
  ```

  In the output of this command, under Non-authoritative answer, look for a record that begins with v=DMARC1. If this record includes the string `aspf=r`, or if the `aspf` string is not present at all, then your domain uses relaxed alignment for SPF. If the record includes the string `aspf=s`, then your domain uses strict alignment for SPF. Your system administrator will need to remove this tag from the DMARC TXT record in your domain's DNS configuration. Alternatively, you can use a web-based DMARC lookup tool, such as the DMARC Inspector from the dmarcian website or the DMARC Check tool from the Proofpoint website, to determine your domain's policy alignment for SPF.

Complying with DMARC through DKIM

For an email to comply with DMARC based on DKIM, both of the following conditions must be met:

- The message must have a valid DKIM signature.
- The From address in the email header must align with the `d=` domain in the DKIM signature. If the domain's DMARC policy specifies strict alignment for DKIM, these domains must match exactly. If the domain's DMARC policy specifies relaxed alignment for DKIM, the `d=` domain can be a subdomain of the From domain.

To comply with these requirements, complete the following steps:

- Set up Easy DKIM by completing the procedures in the section called “Easy DKIM” (p. 131). When you use Easy DKIM, Amazon SES will automatically sign your emails.

  Note
  Rather than use Easy DKIM, you can also manually sign your messages (p. 141). However, you must be very careful if you choose to do so, because Amazon SES does not validate the DKIM signature that you construct. For this reason, we highly recommend using Easy DKIM.
- Ensure that your sending domain uses a relaxed policy for DKIM. If you have not changed your domain's policy alignment, it will use a relaxed policy by default.
Managing your Amazon SES sending quotas

Your Amazon SES account has a set of sending quotas that regulate the number of email messages that you can send and the rate at which you can send them. Sending quotas benefit all Amazon SES customers because they help to maintain the trusted relationship between Amazon SES and email providers. Sending quotas help you to gradually ramp up your sending activity and decrease the likelihood that email providers block your emails because of sudden, unexpected spikes in your email sending volume or rate.

The following quotas apply to sending email through Amazon SES:

- **Maximum daily sends** — The maximum number of emails that you can send in a 24-hour period. This quota is calculated on a rolling time period. Every time you try to send an email, Amazon SES determines the number of emails that you sent in the previous 24 hours. As long as the total number of emails that you have sent in the past 24 hours is less than this daily maximum, your send request is accepted and your email is sent.

  If sending a message would exceed the daily maximum for your account, your call to Amazon SES is rejected.

- **Maximum sending rate** — The maximum number of emails that Amazon SES can accept from your account each second. You can exceed this quota for short bursts, but not for sustained periods of time.

  **Note**
  The rate at which Amazon SES accepts your messages can be less than the maximum send rate for your account.

- **Maximum message size (MB)** — The maximum email size that you can send. This includes any images and attachments that are part of the email after MIME encoding. For example, if you attach a 5MB file, the attachment size in the email after MIME encoding will be ~6.85MB (about 137% of the original file size).

  **Note**
  We recommend you upload your attachments to cloud drives and include the URL of cloud drive attachment to reduce email size and improve deliverability. SES cannot guarantee that large emails will end up in the recipient mailbox as different mail servers will have varying size based policies.

Your Amazon SES sending quotas are separate for each AWS Region. For information about using Amazon SES in multiple AWS Regions, see Regions and Amazon SES (p. 457).
When your account is in the Amazon SES sandbox, you can only send 200 messages per 24-hour period, and your maximum sending rate is one message per second. When you submit a request to have your account removed from the sandbox, you can also request that your quotas are increased at the same time. For more information about having your account removed from the sandbox, see Moving out of the Amazon SES sandbox (p. 71).

When your account has been removed from the sandbox, you can request additional quota increases at any time by creating a new case in the AWS Support Center. For more information, see Increasing your Amazon SES sending quotas (p. 146).

Note
Sending quotas are based on recipients rather than on messages. For example, an email that has 10 recipients counts as 10 against your quota. However, we don't recommend that you send an email to multiple recipients in a single call to the SendEmail API operation, because if the call fails, the entire email is rejected. We recommend that you call SendEmail once for every recipient.

- To increase your sending quotas, see Increasing your Amazon SES sending quotas (p. 146).
- For information about the errors your application receives when you reach your sending quotas, see Errors related to the sending quotas for your Amazon SES account (p. 148).
- To monitor your sending quotas by using the Amazon SES console or the Amazon SES API, see Monitoring your Amazon SES sending quotas (p. 145).

Monitoring your Amazon SES sending quotas

You can monitor your sending quotas by using the Amazon SES console or through the Amazon SES API, whether by calling the Query (HTTPS) interface directly or indirectly through an AWS SDK, the AWS Command Line Interface, or the AWS Tools for Windows PowerShell.

Important
We recommend that you frequently check your sending statistics to ensure that you are not close to your sending quotas. If you are close to your sending quotas, see Increasing your Amazon SES sending quotas (p. 146) for information about how to increase them. Don't wait until you reach your sending quotas to consider increasing them.

Monitoring your sending quotas using the Amazon SES console

The following procedure shows you how to view your sending quotas using the Amazon SES console.

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, choose Sending Statistics. Your sending quotas are shown under Your Amazon SES Sending Limits.
3. To update the display, choose **Refresh**.

**Monitoring your sending quotas using the Amazon SES API**

The Amazon SES API provides the `GetSendQuota` action, which returns your sending quotas. When you call `GetSendQuota` action, you receive the following information:

- Number of emails you have sent during the past 24 hours
- Sending quota for the current 24-hour period
- Maximum send rate

**Note**

For a description of `GetSendQuota`, see *Amazon Simple Email Service API Reference.*

**Increasing your Amazon SES sending quotas**

When your account is out of the sandbox and you're sending high-quality production email, we might automatically increase the sending quotas for your account. Often, we automatically increase these quotas before you actually need them to be increased.

To qualify for automatic rate increases, all of the following statements have to be true:

- **You send high-quality content that your recipients want to receive** – Send content that recipients want and expect. Stop sending email to customers who don't open your email.

- **You send actual production content** – Sending test messages to fake email addresses can have a negative effect on your bounce and complaint rates. Also, sending messages only to internal recipients makes it difficult to determine if you're sending content that customers want to receive. However, when you send your production messages to non-internal recipients, we can accurately assess your email-sending practices.

- **You send near your current quota** – To qualify for an automatic quota increase, your daily email volume should regularly approach the daily maximum for your account without exceeding it.

- **You have low bounce and complaint rates** – Minimize the number of bounces and complaints that you receive. Having a high number of bounces and complaints can have a negative impact on your sending quotas.
If your current sending quotas aren't adequate for your needs and we haven't automatically increased them, you can request an increase. For information about requesting a sending quota increase, see Opening a Case to Increase Amazon SES Sending Quotas (p. 147).

Opening a Case to Increase Amazon SES Sending Quotas

To apply for higher sending quotas for Amazon SES, open a case in Support Center by completing the following steps.

To request higher sending quotas

2. On the Support menu, choose Support Center, as shown in the following image.

3. On the My support cases tab, choose Create case.
4. Under Create case, choose Service limit increase.
5. Under Case classification, complete the following sections:
   - For Limit type, choose SES Service Limits.
   - For Mail Type, choose the type of email that you plan to send. If more than one value applies, choose the option that applies to the majority of the email that you plan to send.
   - For Website URL, enter the URL of your website. Providing this information helps us better understand the type of content that you plan to send.
   - For My email sending complies with the AWS Service Terms and AUP, choose the option that applies to your use case.
   - For I only send to recipients who have specifically requested my mail, choose the option that applies to your use case.
   - For I have a process to handle bounces and complaints, choose the option that applies to your use case.
6. Under Requests, complete the following sections:
   - For Region, choose the AWS Region that your request applies to.
• For **Limit**, choose the type of quota increase that you want to request. You can choose from the following options:

  • **Desired Maximum Send Quota** – Choose this option if you want to request an increase to the number of emails that your account can send per 24-hour period in the selected Region.

  **Desired Maximum Send Rate** – Choose this option if you want to request an increase to the number of emails that your account can send each second in the selected Region.

  • For **New limit value**, enter the quota that you want to increase. Only request the amount that you think you'll need. Remember that you aren't guaranteed to receive the amount that you request.

  **Note**

  If you want to request both a sending quota increase and a sending rate increase, or if you want to request a sending quota increase in a different AWS Region, choose **Add another request**. Then repeat this step.

7. Under **Case Description**, for **Use case description**, describe how you plan to use Amazon SES to send email. To help us process your request, answer the following questions:

  • How do you plan to build or acquire your mailing list?
  • How do you plan to handle bounces and complaints?
  • How can recipients opt out of receiving email from you?
  • How did you choose the new sending rate or sending quota that you specified in this request?

  If there's additional information that we should consider when evaluating your case, provide that information in this section as well.

8. Under **Contact options**, for **Preferred contact language**, choose whether you want to receive communications for this case in **English** or **Japanese**.

9. When you finish, choose **Submit**.

The AWS Support team provides an initial response to your request within 24 hours.

In order to prevent our systems from being used to send unsolicited or malicious content, we have to consider each request carefully. If we're able to do so, we'll grant your request within this 24-hour period. However, if we need to obtain additional information from you, it might take longer to resolve your request.

We might not be able to grant your request if your use case doesn't align with our policies.

**Errors related to the sending quotas for your Amazon SES account**

If you attempt to send an email after reaching your daily sending quota (the maximum amount of email you can send in a 24-hour period) or your maximum sending rate (the maximum number of messages you can send per second), Amazon SES drops the message and doesn't attempt to redeliver it. Amazon SES also provides an error message that explains the issue. The way that Amazon SES produces this error message depends on how you attempted to send the email. This topic includes information about the messages you receive through the Amazon SES API and through the SMTP interface.

For a technique that you can use when you reach your maximum send rate, see **How to handle a "Throttling – Maximum sending rate exceeded" error** on the AWS Messaging and Targeting Blog.
Using sending authorization with Amazon SES

You can configure Amazon SES to authorize other users to send emails from addresses or domains that you own (your identities) using their own Amazon SES accounts. This feature, called *sending authorization*, lets you maintain control over your identities so that you can change or revoke the permissions at any time. For example, if you are a business owner, you can use sending authorization to enable a third party (such as an email marketing company) to send email from a domain you own.

If you want to authorize someone to send emails on your behalf, then you are an *identity owner*. If you are an identity owner, we recommend that you read the following sections:

- Overview of Sending Authorization (p. 150)
- Sending Authorization Policies (p. 152)
- Sending Authorization Policy Examples (p. 156)
- Identity Owner Tasks (p. 161)

If you have been authorized to send emails on behalf of someone else, then you are a *delegate sender*. If you are a delegate sender, we recommend that you read the following sections:

- Overview of Sending Authorization (p. 150)
- Delegate Sender Tasks (p. 166)

**Note**

You can also control access to Amazon SES by using IAM policies. IAM policies constrain what individual IAM users can do, while sending authorization policies constrain how individual verified identities can be used. Further, only sending authorization policies can grant cross-account access. For more information about using IAM policies with Amazon SES, see Controlling access to Amazon SES (p. 397).
Overview of Amazon SES Sending Authorization

This topic provides an overview of the sending authorization process and then explains how the email sending features of Amazon SES, such as sending quotas and notifications, work with sending authorization.

This section uses the following terms:

- **Identity** – An email address or domain that Amazon SES users use to send email.
- **Identity owner** – An Amazon SES user who has verified ownership of an email address or domain by using the procedures described in Verifying identities (p. 47).
- **Delegate sender** – An entity that is authorized to send email from an identity it does not own. An AWS account, an AWS Identity and Access Management (IAM) user, or an AWS service can have this cross-account authority.
- **Sending authorization policy** – A document that you attach to an identity to specify who may send for that identity and under which conditions.
- **Amazon Resource Name (ARN)** – A standardized way to uniquely identify an AWS resource across all AWS services. In the case of sending authorization, the resource is the identity that the identity owner wants the delegate sender to use. An example of an ARN is `arn:aws:ses:us-west-2:123456789012:identity/example.com`.

Sending Authorization Process

Sending authorization is based on sending authorization policies. If you want to enable a delegate sender to send on your behalf, you create a sending authorization policy and associate the policy to your identity by using the Amazon SES console or the Amazon SES API. When the delegate sender attempts to send an email through Amazon SES on your behalf, the delegate sender passes the ARN of your identity in the request or in the header of the email.

When Amazon SES receives the request to send the email, it checks your identity's policy (if present) to determine if you have authorized the delegate sender to send on the identity's behalf. If the delegate sender is authorized, Amazon SES accepts the email; otherwise, Amazon SES returns an error message.

The following diagram shows the high-level relationship between sending authorization concepts:

The sending authorization process consists of the following steps:
1. The identity owner verifies an identity with Amazon SES by using the Amazon SES console or the Amazon SES API. For information about the verification procedure, see Verifying identities (p. 47).

2. The delegate sender gives the identity owner the AWS account ID, IAM user ARN, or AWS service name of the entity that will do the sending.

3. The identity owner creates a sending authorization policy and attaches the policy to the identity by using the Amazon SES console or the Amazon SES API.

4. The identity owner gives the delegate sender the ARN of the identity so that the delegate sender can provide the ARN to Amazon SES at the time of email sending.

5. The delegate sender sets up bounce and complaint notifications. The identity owner can also set up email feedback notifications for bounce and complaint events. Both the identity owner and the delegate sender can also set up event publishing (p. 288) to capture sending event data.

   **Note**
   If the identity owner disables sending event notifications, the delegate sender must set up event publishing to publish bounce and complaint events to an Amazon SNS topic or a Kinesis Data Firehose stream. The sender must also apply the configuration set that contains the event publishing rule to each email they send. If neither the identity owner nor the delegate sender sets up a method of sending notifications for bounce and complaint events, then Amazon SES automatically sends event notifications by email to the address in the Return-Path field of the email (or the address in the Source field, if you didn't specify a Return-Path address), even if the identity owner disabled email feedback forwarding.

6. The delegate sender attempts to send an email through Amazon SES on behalf of the identity owner by passing the ARN of the identity owner's identity in the request or in the header of the email. The delegate sender can send the email by using either the Amazon SES SMTP interface or the Amazon SES API. Upon receiving the request, Amazon SES examines any policies that are attached to the identity, and accepts the email if the delegate sender is authorized to use the specified "From" address and "Return Path" address; otherwise, Amazon SES returns an error and does not accept the message.

7. If the identity owner needs to de-authorize the delegate sender, the identity owner edits the sending authorization policy or deletes the policy entirely. The identity owner can perform either action by using the Amazon SES console or the Amazon SES API.

For more information about how the identity owner or delegate sender perform those tasks, see Identity Owner Tasks (p. 161) or Delegate Sender Tasks (p. 166), respectively.

**Attribution of Email Sending Features**

It is important to understand the role of the delegate sender and the identity owner with respect to Amazon SES email sending features such as daily sending quota, bounces and complaints, DKIM signing, feedback forwarding, and so on. The attribution is the following:

- **Sending quotas** – Email sent from the identity owner’s identities count against the delegate sender’s quotas.

- **Bounces and complaints** – Bounce and complaint events are recorded against the delegate sender's Amazon SES account, and can therefore impact the delegate sender's reputation.

- **DKIM signing** – If the identity owner has enabled Easy DKIM signing for an identity, all email sent from that identity will be DKIM-signed, including email sent by the delegate sender. Only the identity owner can control whether the emails are DKIM-signed.

- **Notifications** – Both the identity owner and the delegate sender can set up notifications for bounces and complaints. The email identity owner can also enable email feedback forwarding. For information about setting up notifications, see Monitoring your Amazon SES sending activity (p. 259).

- **Verification** – Identity owners are responsible for following the procedure in Verifying identities (p. 47) to verify that they own the email addresses and domains that they are authorizing delegate senders to use. Delegate senders do not need to verify any email addresses or domains specifically for sending authorization.
• **AWS Regions** – The delegate sender must send the emails from the AWS Region in which the identity owner’s identity is verified. The sending authorization policy that gives permission to the delegate sender must be attached to the identity in that region.

• **Billing** – All messages that are sent from the delegate sender's account, including emails that the delegate sender sends using the identity owner's addresses, are billed to the delegate sender.

## Amazon SES Sending Authorization Policies

To enable another AWS account, Identity Access and Management (IAM) user, or AWS service to send email through Amazon SES on your behalf, you create a sending authorization policy, which is a JSON document that you attach to an identity that you own. The policy explicitly lists who you're allowing to send for that identity, and under which conditions. All senders except you and the entities that you explicitly grant permissions to in the policies are denied. An identity can have no policy, one policy, or multiple policies attached to it. You can also have one policy with multiple statements to achieve the effect of multiple policies.

Policies can be simple, or can be configured to provide fine-grained control. For example, if you owned example.com, you could write a simple policy to grant AWS ID 123456789012 permission to send from that domain. A more detailed policy could specify that AWS ID 123456789012 can send email only from user@example.com and only within a specified date range.

Amazon SES sending authorization policies apply to email sending APIs (SendEmail, SendRawEmail, SendTemplatedEmail, and SendBulkTemplatedEmail) only. They don't enable a user to access your AWS account in any other way.

### Policy Structure

Each sending authorization policy is a JSON document that is attached to an identity. Each policy includes the following sections:

• Policy-wide information at the top of the document.

• One or more individual statements, each of which describes a set of permissions.

The following example policy grants AWS account ID 123456789012 permission to send from the verified domain example.com.

```json
{
   "Id":"ExampleAuthorizationPolicy",
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid":"AuthorizeAccount",
         "Effect":"Allow",
         "Principal": {
            "AWS": [ "123456789012"
            ],
         },
         "Action": [ "ses:SendEmail", "ses:SendTemplatedEmail", "ses:SendRawEmail", "ses:SendBulkTemplatedEmail" ]
      }
   ]
}
```
You can find more sending authorization policy examples at Sending Authorization Policy Examples (p. 156).

Policy Elements

This section describes the elements contained in sending authorization policies. First we describe policy-wide elements, and then we describe elements that apply only to the statement in which they are included. We follow with a discussion of how to add conditions to your statements.

For specific information about the syntax of the elements, see Grammar of the IAM Policy Language in the IAM User Guide.

Policy-Wide Information

There are two policy-wide elements: Id and Version. The following table provides information about these elements.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Required</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Uniquely identifies the policy.</td>
<td>No</td>
<td>Any string</td>
</tr>
<tr>
<td>Version</td>
<td>Specifies the policy access language version.</td>
<td>No</td>
<td>Any string. As a best practice, we recommend that you include this field with a value of &quot;2012-10-17&quot;.</td>
</tr>
</tbody>
</table>

Statements Specific to the Policy

Sending authorization policies require at least one statement. Each statement can include the elements described in the following table.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Required</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sid</td>
<td>Uniquely identifies the statement.</td>
<td>No</td>
<td>Any string</td>
</tr>
<tr>
<td>Effect</td>
<td>Specifies the result that you want the policy statement to return at evaluation time.</td>
<td>Yes</td>
<td>&quot;Allow&quot; or &quot;Deny&quot;.</td>
</tr>
<tr>
<td>Resource</td>
<td>Specifies the identity to which the policy applies. This is the email address or domain that the identity owner is authorizing the delegate sender to use.</td>
<td>Yes</td>
<td>The Amazon Resource Name (ARN) of the email identity.</td>
</tr>
<tr>
<td>Principal</td>
<td>Specifies the AWS account, IAM user,</td>
<td>Yes</td>
<td>A valid AWS account ID, IAM user ARN, or</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td>Required</td>
<td>Valid Values</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Name                      | or AWS service that receives the permission in the statement.               |          | AWS service. AWS account IDs and IAM user ARNs are specified using "AWS" (for example, "AWS": ["123456789012"]) or "AWS": ["arn:aws:iam::123456789012:root"].
AWS service names are specified using "Service" (for example, "Service": ["cognito-idp.amazonaws.com"]).
For examples of the format of IAM user ARNs, see the AWS General Reference. |
| Action                    | Specifies the email sending action that the statement applies to.           | Yes      | "ses:SendEmail", "ses:SendRawEmail", "ses:SendTemplatedEmail", "ses:SendBulkTemplatedEmail"
You can specify one or more of these operations. You can also specify "ses:Send*" to encompass all of these operations. If the delegate sender plans to send email by using the SMTP interface, you have to specify "ses:SendEmail", or use "ses:Send*". |
| Condition                 | Specifies any restrictions or details about the permission.                 | No       | See the information about conditions following this table.                                                                                |

**Conditions**

A *condition* is any restriction about the permission in the statement. The part of the statement that specifies the conditions can be the most detailed of all the parts. A *key* is the specific characteristic that's the basis for access restriction, such as the date and time of the request.

You use both conditions and keys together to express the restriction. For example, if you want to restrict the delegate sender from making requests to Amazon SES on your behalf after July 30, 2019, you use the condition called `DateLessThan`. You use the key called `aws:CurrentTime` and set it to the value `2019-07-30T00:00:00Z`.

You can use any of the AWS-wide keys listed at Available Keys in the IAM User Guide, or you can use one of the following keys specific to Amazon SES:
### Condition Key

<table>
<thead>
<tr>
<th>Condition Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ses:Recipients</td>
<td>Restricts the recipient addresses, which include the To:, &quot;CC&quot;, and &quot;BCC&quot; addresses.</td>
</tr>
<tr>
<td>ses:FromAddress</td>
<td>Restricts the &quot;From&quot; address.</td>
</tr>
<tr>
<td>ses:FromDisplayName</td>
<td>Restricts the contents of the string that is used as the &quot;From&quot; display name (sometimes called &quot;friendly from&quot;). For example, the display name of &quot;John Doe <a href="mailto:johndoe@example.com">johndoe@example.com</a>&quot; is John Doe.</td>
</tr>
<tr>
<td>ses:FeedbackAddress</td>
<td>Restricts the &quot;Return Path&quot; address, which is the address where bounce and complaints can be sent to you by email feedback forwarding. For information about email feedback forwarding, see Amazon SES notifications sent by email (p. 265).</td>
</tr>
</tbody>
</table>

You can use the `StringEquals` and `StringLike` conditions with Amazon SES keys. These conditions are for case-sensitive string matching. For `StringLike`, the values can include a multi-character match wildcard (*) or a single-character match wildcard (?) anywhere in the string. For example, the following condition specifies that the delegate sender can only send from a "From" address that starts with *invoicing* and ends with *@example.com*:

```
"Condition": {
    "StringLike": {
        "ses:FromAddress": "invoicing*@example.com"
    }
}
```

You can also use the `StringNotLike` condition to prevent delegate senders from sending email from certain email addresses. For example, you can disallow sending from `admin@example.com`, as well as similar addresses such as "admin"@example.com, admin+1@example.com, or `sender@admin.example.com`, by including the following condition in your policy statement:

```
"Condition": {
    "StringNotLike": {
        "ses:FromAddress": "*admin*example.com"
    }
}
```

For more information about how to specify conditions, see IAM JSON Policy Elements: Condition in the IAM User Guide.

## Policy Requirements

Policies must meet all of the following requirements:

- Each policy has to include at least one statement.
- Each policy has to include at least one valid principal.
- Each policy has to specify one resource, and that resource has to be the ARN of the identity that the policy is attached to.
- Identity owners can associate up to 20 policies with each unique identity.
- Policies can’t exceed 4 kilobytes (KB) in size.
Policy names can't exceed 64 characters. Additionally, they can only include alphanumeric characters, dashes, and underscores.

Amazon SES Sending Authorization Policy Examples

Sending authorization enables you to specify the fine-grained conditions under which you allow delegate senders to send on your behalf.

The following examples show you how to write policies to control different aspects of sending:

- Specifying the Delegate Sender (p. 156)
- Restricting the "From" Address (p. 158)
- Restricting the Time at which the Delegate can Send Email (p. 158)
- Restricting the Email Sending Action (p. 159)
- Restricting the Display Name of the Email Sender (p. 159)
- Using Multiple Statements (p. 160)

Specifying the Delegate Sender

The principal, which is the entity to which you are granting permission, can be an AWS account, an AWS Identity and Access Management (IAM) user, or an AWS service.

The following example shows a simple policy that allows AWS ID 123456789012 to send email from the verified identity example.com (which is owned by AWS account 888888888888). The Condition statement in this policy only allows the delegate (that is, AWS ID 123456789012) to send email from the address marketing+.*@example.com, where .* is any string that the sender wants to add after marketing+

```json
{
    "Id":"SampleAuthorizationPolicy",
    "Version":"2012-10-17",
    "Statement":[
        {
            "Sid":"AuthorizeMarketer",
            "Effect":"Allow",
            "Principal":{
                "AWS": [
                    "123456789012"
                ]
            },
            "Action": [
                "SES:SendEmail",
                "SES:SendRawEmail"
            ],
            "Condition":{
                "StringLike":{
                    "ses:FromAddress":"marketing+.*@example.com"
                }
            }
        }
    ]
}
```

The following example policy grants permission to two IAM users to send from identity example.com. IAM users are specified by their Amazon Resource Name (ARN).

```json
{
}
```
The following example policy grants permission to Amazon Cognito to send from identity `example.com`.

```json
{
    "Id":"ExampleAuthorizationPolicy",
    "Version":"2012-10-17",
    "Statement":[
        {
            "Sid":"AuthorizeService",
            "Effect":"Allow",
            "Principal":{
                "Service":{
                    "cognito-idp.amazonaws.com"
                },
                "Action":[
                    "SES:SendEmail",
                    "SES:SendRawEmail"
                ],
                "Condition": {
                    "StringEquals": {
                        "aws:SourceAccount": "888888888888",
                        "aws:SourceArn": "arn:aws:cognito-idp:us-east-1:888888888888:userpool/your-user-pool-id-goes-here"
                    }
                }
            }
        }
    ]
}
```

The following example policy grants permission to all accounts within an AWS Organization to send from identity `example.com`. The AWS Organization is specified using the `PrincipalOrgID` global condition key.

```json
{
    "Id":"ExampleAuthorizationPolicy",
    "Version":"2012-10-17",
    "Statement":[
        {
            "Sid":"AuthorizeOrg",
            "Effect":"Allow",
            "Principal":{"*"}
        }
    ]
}
```
Restricting the "From" Address

If you use a verified domain, you may want to create a policy that only allows the delegate sender to send from a specified email address. To restrict the "From" address, you set a condition on the key called `ses:FromAddress`. The following policy enables AWS account ID 123456789012 to send from the identity `example.com`, but only from the email address `sender@example.com`.

```
"Action": [
    "SES:SendEmail",
    "SES:SendRawEmail"
],
"Condition": {
    "StringEquals": {
        "aws:PrincipalOrgID": "o-xxxxxxxxxx"
    }
}
```

Restricting the Time at which the Delegate can Send Email

You can also configure your sender authorization policy so that a delegate sender can only send email at a certain time of day, or within a certain date range. For example, if you plan to send an email campaign during the month of September 2018, you can use the following policy to restrict the delegate's ability to send email to that month only.

```
"Action": [
    "SES:SendEmail",
    "SES:SendRawEmail"
],
"Condition": {
    "StringEquals": {
        "ses:FromAddress": "sender@example.com"
    }
}
```

```
"Id": "ExamplePolicy",
"Version": "2012-10-17",
"Statement": [
    {
        "Sid": "AuthorizeFromAddress",
        "Effect": "Allow",
        "Principal": {
            "AWS": [
                "123456789012"
            ]
        },
        "Action": [
            "SES:SendEmail",
            "SES:SendRawEmail"
        ],
        "Condition": {
            "StringEquals": {
                "ses:FromAddress": "sender@example.com"
            }
        }
    }
]
```
Restricting the Email Sending Action

There are two actions that senders can use to send an email with Amazon SES: **SendEmail** and **SendRawEmail**, depending on how much control the sender wants over the format of the email. Sending authorization policies enable you to restrict the delegate sender to one of those two actions. However, many identity owners leave the details of the email sending calls up to the delegate sender by enabling both actions in their policies.

**Note**

If you want to enable the delegate sender to access Amazon SES through the SMTP interface, you must choose **SendRawEmail** at a minimum.

If your use case is such that you want to restrict the action, you can do so by including only one of the actions in your sending authorization policy. The following example shows you how to restrict the action to **SendRawEmail**.

```json
{
  "Id":"ExamplePolicy",
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"ControlAction",
      "Effect":"Allow",
      "Principal":{
        "AWS":[
          "123456789012"
        ]
      },
      "Action":[
        "SES:SendRawEmail"
      ],
      "Condition":{
        "DateGreaterThan":{
          "aws:CurrentTime":"2018-08-31T12:00Z"
        },
        "DateLessThan":{
          "aws:CurrentTime":"2018-10-01T12:00Z"
        }
      }
    }
  ]
}
```

Restricting the Display Name of the Email Sender

Some email clients display the "friendly" name of the email sender (if the email header provides it), rather than the actual "From" address. For example, the display name of "John Doe <johndoe@example.com>" is John Doe. For instance, you might send emails from user@example.com,
but you prefer that recipients see that the email is from Marketing rather than from user@example.com. The following policy enables AWS account ID 123456789012 to send from identity example.com, but only if the display name of the "From" address includes Marketing.

```
{
  "Id":"ExamplePolicy",
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"AuthorizeFromAddress",
      "Effect":"Allow",
      "Principal":{
        "AWS": [
          "123456789012"
        ]
      },
      "Action": [
        "SES:SendEmail",
        "SES:SendRawEmail"
      ],
      "Condition":{
        "StringLike":{
          "ses:FromDisplayName":"Marketing"
        }
      }
    }
  ]
}
```

### Using Multiple Statements

Your sending authorization policy can include multiple statements. The following example policy has two statements. The first statement authorizes two AWS accounts to send from sender@example.com as long as the "From" address and the feedback address both use the domain example.com. The second statement authorizes an IAM user to send email from sender@example.com as long as the recipient's email address is under the example.com domain.

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"AuthorizeAWS",
      "Effect":"Allow",
      "Resource":"arn:aws:ses:us-east-1:999999999999:identity/sender@example.com",
      "Principal":{
        "AWS": [
          "111111111111",
          "222222222222"
        ]
      },
      "Action": [
        "SES:SendEmail",
        "SES:SendRawEmail"
      ],
      "Condition":{
        "StringLike":{
          "ses:FromAddress":"*@example.com",
          "ses:FeedbackAddress":"*@example.com"
        }
      }
    },
    {
      "Sid":"AuthorizeEmail",
      "Effect":"Allow",
      "Principal":{
        "AWS": [
          "123456789012"
        ]
      },
      "Action": ["SES:SendEmail", "SES:SendRawEmail"],
      "Condition":{
        "StringLike":{
          "ses:FromEmailAddress":"*@example.com",
          "ses:FeedbackEmailAddress":"*@example.com"
        }
      }
    }
  ]
}
```
Identity Owner Tasks for Amazon SES Sending Authorization

This section describes the steps that identity owners must take when configuring sending authorization.

Topics

- Verifying an Identity for Amazon SES Sending Authorization (p. 161)
- Setting Up Identity Owner Notifications for Amazon SES Sending Authorization (p. 161)
- Getting Information from the Delegate Sender for Amazon SES Sending Authorization (p. 162)
- Creating a Policy for Amazon SES Sending Authorization (p. 162)
- Providing the Delegate Sender with the Identity Information for Amazon SES Sending Authorization (p. 165)
- Managing Your Policies for Amazon SES Sending Authorization (p. 165)

Verifying an Identity for Amazon SES Sending Authorization

The first step in configuring sending authorization is to prove that you own the email address or domain that the delegate sender will use to send email. The verification procedure is described in Verifying identities (p. 47).

You can confirm that an email address or domain is verified by checking its status in the Identity Management section of the Amazon SES console or by using the GetIdentityVerificationAttributes API operation.

Before you or the delegate sender can send email to non-verified email addresses, you have to submit a request to have your account removed from the Amazon SES sandbox. For more information, see Moving out of the Amazon SES sandbox (p. 71).

Important
The AWS accounts of both the identity owner and the delegate sender have to be removed from the sandbox before either account can send email to non-verified addresses.

Setting Up Identity Owner Notifications for Amazon SES Sending Authorization

If you authorize a delegate sender to send email on your behalf, Amazon SES counts all bounces or complaints that those emails generate toward the delegate sender's bounce and complaint limits, rather
than your own. However, if your IP addresses end up on 3rd-party anti-spam DNS-based Blackhole Lists (DNSBLs) as a result of messages sent by a delegate sender, the reputation of your identities may be damaged. For this reason, if you’re an identity owner, you should set up email feedback forwarding for your identities. For more information, see Amazon SES notifications sent by email (p. 265).

Delegate senders can set up their own bounce and complaint notifications for the identities that you have authorized them to use. They can also set up event publishing (p. 288) to send notifications when bounce or complaint events occur. If the identity owner disables feedback forwarding, the delegate sender must set up event publishing to publish bounce and complaint events to an Amazon SNS topic or a Kinesis Data Firehose stream. If neither the identity owner nor the delegate sender sets up a method of sending notifications for bounce and complaint events, or if the sender doesn’t apply the configuration set that uses the event publishing rule, then Amazon SES automatically sends event notifications by email to the address in the Return-Path field of the email (or the address in the Source field, if you didn’t specify a Return-Path address), even if you disabled email feedback forwarding. This process is illustrated in the following image.

Getting Information from the Delegate Sender for Amazon SES Sending Authorization

Your sending authorization policy must specify at least one principal, which is the entity to which you are granting access. For Amazon SES sending authorization policies, the principal can be an AWS account, an AWS Identity and Access Management (IAM) user, or an AWS service.

The type of principal you choose depends on your preference, but if you want the finest grain control, ask the delegate sender to set up an IAM user so that only one delegate sender can send for you rather than any user in the delegate sender’s AWS account. The delegate sender can find information about setting up an IAM user in Creating an IAM User in Your AWS Account in the IAM User Guide.

After you have decided whether you want to grant access to an AWS account, an IAM user, or an AWS service, ask the delegate sender for the AWS account ID or the IAM user’s Amazon Resource Name (ARN) so that you can include it in your sending authorization policy. You can refer your delegate sender to the instructions for finding this information in Providing Information to the Identity Owner (p. 167). If the delegate sender is an AWS service, see the documentation for that service to determine the service name.

Creating a Policy for Amazon SES Sending Authorization

To authorize a delegate sender to send emails using an email address or domain (an identity) that you own, you create a sending authorization policy, and then attach that policy to the identity. An identity can have zero, one, or many policies. However, a single policy can only be associated with a single identity.

You can create a sending authorization policy in the following ways:

- **By using the Policy Generator** – You can create a simple policy by using the Policy Generator in the Amazon SES console. In addition to specifying who can send the emails, you can constrain the email-sending with conditions based on the time and date range in which emails can be sent, the "From" address, the "From" display name, the address to which bounces and complaints are sent, the recipient addresses, and the source IP. You might also want to use the Policy Generator to create the structure of a simple policy and then customize it later by editing the policy.

- **By creating a Custom Policy** – If you want to include more advanced conditions or use an AWS service as the principal, you can create a custom policy and attach it to the identity by using the Amazon SES console or the Amazon SES API.

This topic describes both methods.
Identity Owner Tasks

Note
Sending authorization policies that you attach to email address identities take precedence over policies that you attach to their corresponding domain identities. For example, if you create a policy for example.com that disallows a delegate sender, and you create a policy for sender@example.com that allows the delegate sender, then the delegate sender can send email from sender@example.com, but not from any other address on the example.com domain.
If you create a policy for example.com that allows a delegate sender, and you create a policy for sender@example.com that disallows the delegate sender, then the delegate sender can send email from any address on the example.com domain, except for sender@example.com.

Creating a Policy Using the Policy Generator

You can use the Policy Generator to create a simple authorization policy by using the following procedure.

To create a policy by using the Policy Generator

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under Identity Management, choose either Domains or Email Addresses.
3. In the list of identities, choose the identity for which you want to create a policy.
4. In the details pane, expand Identity Policies, choose Create Policy, and then choose Policy Generator.
5. In the wizard, create a policy statement by choosing values for the following fields. You can find information about these options in Sending Authorization Policies (p. 152).
   • Effect – If you want to grant access, choose Allow; otherwise, choose Deny.
   • Principals – Enter either the 12-digit AWS account ID or the ARN of an IAM user that you are allowing or denying access, and then choose Add. You can add more principals by repeating this step. An example of an AWS account ID is 123456789012 and an example of an IAM user ARN is arn:aws:iam::123456789012:user/John.
     Note
     The policy generator wizard does not currently support AWS service principals. To add an AWS service principal, you must either create a custom policy (p. 164) or use the policy generator to add an AWS account or IAM user principal, and then edit (p. 165) the policy.
   • Actions – Choose the email-sending access to which this policy applies. Typically, identity owners choose both options to give the delegate sender the choice to choose how to implement the email sending. For more information, see Statements Specific to the Policy (p. 153).
6. (Optional) If you want to add restrictions to the policy, choose Add Conditions, and then choose the following information:
   • Key – This is the characteristic that is the basis for access restriction. The Policy Generator lets you choose an Amazon SES-specific key or one of a few commonly used AWS-wide keys (current time and source IP). For details, see Conditions (p. 154). If you want to specify the more advanced AWS-wide keys listed in Available Keys, you can edit the policy after you create it.
   • Condition – This is the type of condition that you want to specify. For example, there are string conditions, numeric conditions, date and time conditions, and so on. For a list of conditions, see Condition Types in the IAM User Guide.
   • Value – This is the value that will be tested against the condition. For examples, see the policies in Sending Authorization Policy Examples (p. 156).
After you choose the key, condition, and value, choose **Add Condition**. The condition appears in the **Conditions** list. You can remove conditions by choosing **Remove** next to a condition in the list. You can add another condition by choosing **Add Conditions** again.

7. When you finish adding conditions, choose **Add Statement**. The statement appears in the **Statements** list, where you can choose to edit or remove it. You can add additional statements by repeating steps 5–7.

8. When you finish adding statements, choose **Next**.

9. In the **Edit Policy** dialog box, review your policy, edit it if necessary, and then choose **Apply Policy**.

---

**Creating a Custom Policy**

If you want to create a custom policy and attach it to an identity, you have the following options:

- **Using the Amazon SES API** – Create a policy in a text editor and then attach the policy to the identity by using the **PutIdentityPolicy** API described in the Amazon Simple Email Service API Reference.

- **Using the Amazon SES console** – Create a policy in a text editor and attach it to an identity by pasting it into the Custom Policy editor in the Amazon SES console. The following procedure describes this method.
To create a custom policy by using the Custom Policy editor

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under Identity Management, choose either Domains or Email Addresses.
3. In the list of identities, choose the identity for which you want to create a policy.
4. In the details pane, expand Identity Policies, choose Create Policy, and then choose Custom Policy.
5. In the Edit Policy pane, paste the text of your policy.
6. Choose Apply Policy.

Providing the Delegate Sender with the Identity Information for Amazon SES Sending Authorization

After you create your sending authorization policy and attach it to your identity, you can provide the delegate sender with the Amazon Resource Name (ARN) of the identity. The delegate sender will pass that ARN to Amazon SES in the email-sending operation or in the header of the email. Use the following procedure to find your identity's ARN.

To find the ARN of an identity

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under Identity Management, choose either Domains or Email Addresses.
3. In the list of identities, choose the identity to which you attached the sending authorization policy.
4. At the top of the details pane, after Identity ARN, you will see the identity's ARN. It will look similar to arn:aws:ses:us-east-1:123456789012:identity/user@example.com. Copy the entire ARN and give it to your delegate sender.

Managing Your Policies for Amazon SES Sending Authorization

In addition to creating and attaching policies to identities as explained in Creating a Policy (p. 162), you can edit, remove, list, and retrieve an identity's policies, as described in the following sections.

Note
To revoke permissions, you can either edit a policy or remove it.

Editing a Policy

The easiest way to edit a policy is to use the Amazon SES console. If you want to use the Amazon SES API instead, you can use the GetIdentityPolicies operation to retrieve the policy, edit the policy using a text editor, and then use the PutIdentityPolicy operation to overwrite the older policy.

The following procedure shows you how to edit a policy by using the Amazon SES console.

To edit a policy by using the Amazon SES console

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under Identity Management, choose either Domains or Email Addresses.
3. In the list of identities, choose the identity that is associated with the policy that you want to edit.
4. In the details pane, expand **Identity Policies**.
5. Next to the policy that you want to edit, choose **Edit Policy**.
6. In the **Edit Policy** pane, edit the policy, and then choose **Apply Policy**.
7. In the **Overwrite Existing Policy** dialog box, choose **Overwrite**.

**Removing a Policy**

To revoke permissions at any time, you can simply remove the policy. You can remove a policy by using the **DeleteIdentityPolicy** API operation, or you can use the Amazon SES console, as described in the following procedure.

**Important**
After you remove a policy, there is no way to get it back. We recommend that you back up the policy by copying and pasting it into a text file before you remove the policy.

**To remove a policy by using the Amazon SES console**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under **Identity Management**, choose either **Domains** or **Email Addresses**.
3. In the list of identities, choose the identity that is associated with the policy that you want to remove.
4. In the details pane, expand **Identity Policies**. Next to the policy that you want to remove, choose **Remove Policy**.
5. In the **Remove Policy** dialog box, choose **Yes, Remove Policy**.

**Listing and Retrieving Policies**

You can list the policies that are attached to an identity by using the **ListIdentityPolicies** API operation. You can also retrieve the policies themselves by using the **GetIdentityPolicies** API operation.

You can also use the Amazon SES console to perform both of these tasks, as described in the following procedure.

**To list and show the policies attached to an identity by using the Amazon SES console**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under **Identity Management**, choose either **Domains** or **Email Addresses**.
3. In the list of identities, choose the identity for which you want to see policies.
4. In the details pane, expand **Identity Policies**.
5. Next to the policy that you want to view, choose **Show Policy**.

**Delegate Sender Tasks for Amazon SES Sending Authorization**

As a delegate sender, you are sending cross-account emails. This means that you are sending emails on behalf of an identity that you do not own, but are authorized to use. Even though you are sending on the identity owner’s behalf, bounces and complaints count toward the bounce and complaint metrics for
Delegate Sender Tasks

Your AWS account, and the number of messages you send counts toward your sending quota. You are also responsible for requesting any sending quota increases that you might need in order to send the identity owner's emails.

As a delegate sender, you must complete the following tasks:

- Providing Information to the Identity Owner (p. 167)
- Using Delegate Sender Notifications (p. 167)
- Sending Emails for the Identity Owner (p. 170)

Providing Information to the Identity Owner for Amazon SES Sending Authorization

As a delegate sender, you must provide the identity owner with your AWS account ID or the Amazon Resource Name (ARN) of the AWS Identity and Access Management (IAM) user who will send email on behalf of the identity owner. You can find this information by using the following procedures.

To find your AWS account ID

2. In the navigation menu, choose your name, and then choose My Account.
3. Expand Account Settings. This section displays your AWS account ID.

To find the ARN of an IAM user

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose Users.
3. In the list of users, choose the user name. The Summary section displays the ARN. The ARN resembles the following example: arn:aws:iam::123456789012:user/John.

Using Delegate Sender Notifications for Amazon SES Sending Authorization

As the delegate sender, bounces and complaints count toward your bounce and complaint metrics, not those of the identity owner. If the bounce or complaint rates for your account get too high, we might place your account under review or pause your account's ability to send email. For this reason, it's important that you set up notifications and have a process in place to monitor them. You also need to have a process in place for removing addresses that have bounced or complained from your mailing lists.

Delegate senders can set up Amazon SES to send notifications when bounce and complaint events occur. Delegate senders can also set up event publishing (p. 288) to publish bounce and complaint notifications to Amazon SNS or Kinesis Data Firehose.

Note
If you set up Amazon SES to send notifications by using Amazon SNS, you're charged standard Amazon SNS rates for the notifications you receive. For more information, see the Amazon SNS pricing page.

Topics
- Setting Up an Amazon SES Cross-Account Identity Notification Configuration (p. 168)
- Editing an Amazon SES Cross-Account Notification Configuration (p. 168)
• Viewing Your Amazon SES Cross-Account Identity Notifications (p. 169)
• Removing an Amazon SES Cross-Account Identity Notification Configuration (p. 169)

Setting Up an Amazon SES Cross-Account Identity Notification Configuration

Before you set up notifications, you need to know the Amazon Resource Name (ARN) of the identity that the identity owner has authorized you to use, and for which you want to configure notifications. For example, the ARN for identity user@example.com would look similar to arn:aws:ses:us-east-1:123456789012:identity/user@example.com. If the identity owner has not given you the identity's ARN, refer them to the procedure in Providing the Delegate Sender with the Identity Information (p. 165).

The easiest way to configure notifications is to use the Amazon SES console. You can also use the SetIdentityNotificationTopic API operation, passing the identity's ARN as the Identity parameter.

The following procedure shows you how to set up notifications by using the Amazon SES console.

To set up Amazon SNS bounce, complaint, or delivery notifications by using the Amazon SES console

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, choose Cross-Account Notifications.
3. Choose Add Notification Config.
4. On the Add Notification Configuration dialog box, for Identity ARN, type the ARN of the identity that you want to configure notifications for. The identity can't belong to the account that you're currently logged in to.
5. Select the Amazon SNS topics that you want to use for bounces, complaints, or deliveries. You can also create new Amazon SNS topics for these notifications.
   
   Important
   The Amazon SNS topics that you use for Amazon SES notifications must be in the same AWS Region that you use for sending email using Amazon SES.

   You can choose to publish bounce, complaint, and delivery notifications to the same Amazon SNS topic or to different Amazon SNS topics. If you want to use an Amazon SNS topic that you do not own, then the owner of that topic must configure an Amazon SNS access policy that allows your account to call the SNS:Publish action on their topic. For information about how to control access to your Amazon SNS topic through the use of IAM policies, see Managing Access to Your Amazon SNS Topics.

6. Choose Save Config to save your notification configuration. There may be a brief delay before these changes take effect.

Editing an Amazon SES Cross-Account Notification Configuration

The easiest way to edit notification configurations is to use the Amazon SES console. If you want to use the Amazon SES API instead, you can use the SetIdentityNotificationTopic API operation and pass the identity's ARN as the Identity parameter.

The following procedure shows you how to edit a cross-account notification configuration by using the Amazon SES console.

To edit a cross-account notification configuration by using the Amazon SES console

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, choose **Cross-Account Notifications**.

   The cross-account identities for which you have set up notifications are listed in the **Cross-Account Notifications** details pane.

3. Choose the ARN of the identity for which you want to view the notification configuration.

4. Edit the notification settings, and then choose **Save Config**.

**Viewing Your Amazon SES Cross-Account Identity Notifications**

The easiest way to view your notification configurations is to use the Amazon SES console. You can also use the `GetIdentityNotificationAttributes` API operation, passing the identity's ARN as the `Identity` parameter.

*Note*

The only cross-account identities displayed in the cross-account identity list are the identities for which you have configured notifications by using the procedure described in [Setting Up a Notification Configuration](#p-168).

**To view your cross-account notification configurations by using the Amazon SES console**

1. Sign in to the AWS Management Console and open the Amazon SES console at `https://console.aws.amazon.com/ses/`.

2. In the left navigation pane, choose **Cross-Account Notifications**.

   The cross-account identities for which you have set up notifications are listed in the **Cross-Account Notifications** details pane.

3. Choose the ARN of an identity.

   The **Edit Configuration Notification** dialog box displays the identity's settings.

**Removing an Amazon SES Cross-Account Identity Notification Configuration**

The easiest way to remove a notification configuration is to use the Amazon SES console. You can also use the `SetIdentityNotificationTopic` API operation, passing the identity's ARN as the `Identity` parameter, and passing null for the `SnsTopic` parameter. To completely remove the notification configuration, you must perform this operation for each type of notification type (bounce, complaint, or delivery) that was set.

*Note*

When you remove a notification configuration, the ARN of the cross-account identity is removed from the list of cross-account identity ARNs in the Amazon SES console. This does not mean that you cannot continue to send for that identity; it just means that you are no longer set up to receive bounce, complaint, or delivery notifications for it. If you want to re-enable notifications, you need to repeat the notification setup procedure described in [Setting Up a Notification Configuration](#p-168).

The following procedure shows you how to remove a cross-account notification configuration by using the Amazon SES console.

**To remove a cross-account notification configuration by using the Amazon SES console**

1. Sign in to the AWS Management Console and open the Amazon SES console at `https://console.aws.amazon.com/ses/`.

2. In the left navigation pane, choose **Cross-Account Notifications**.

   The cross-account identities for which you have set up notifications are listed in the **Cross-Account Notifications** details pane.
3. Choose the box to the left of the cross-identity that you want to remove, and then choose Remove.
4. In the Remove Cross-Account Notification Config dialog box, choose Delete Notification config.

The ARN of the cross-account identity no longer appears in the list of cross-account identity x. This does not mean that you cannot send for the identity, just that you no longer have configured notifications for it.

Sending Emails for the Identity Owner for Amazon SES Sending Authorization

As a delegate sender, you send emails the same way that other Amazon SES senders do, except that you provide the ARN of the identity that the identity owner has authorized you to use. When you call Amazon SES to send the email, Amazon SES checks to see if the identity that you specified has a policy that authorizes you to send for it.

There are different ways that you can specify the identity's ARN when you send an email. The method that you can use depends on whether you send the email by using the Amazon SES API operations or the Amazon SES SMTP interface.

**Important**
To successfully send an email, you have to connect to the Amazon SES endpoint in the AWS Region that the identity owner verified the identity in. Additionally, the AWS accounts of both the identity owner and the delegate sender have to be removed from the sandbox before either account can send email to non-verified addresses. For more information, see Moving out of the Amazon SES sandbox (p. 71).

Using the Amazon SES API

As with any Amazon SES email sender, if you access Amazon SES through the Amazon SES API (either directly through HTTPS or indirectly through an AWS SDK), you can choose between one of two email-sending actions: SendEmail and SendRawEmail. The Amazon Simple Email Service API Reference describes the details of these APIs, but we provide an overview of the sending authorization parameters here.

SendRawEmail

If you want to use SendRawEmail so that you can control the format of your emails, you can specify the cross-account identity in one of two ways:

- **Pass optional parameters to the SendRawEmail API.** The required parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SourceArn</td>
<td>The ARN of the identity that is associated with the sending authorization policy that permits you to send for the email address specified in the Source parameter of SendRawEmail.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong></td>
</tr>
<tr>
<td></td>
<td>If you only specify the SourceArn, Amazon SES sets the &quot;From&quot; address and the &quot;Return Path&quot; addresses to the identity that you specified in SourceArn.</td>
</tr>
<tr>
<td>FromArn</td>
<td>The ARN of the identity that is associated with the sending authorization policy that permits you to specify</td>
</tr>
</tbody>
</table>
### Delegate Sender Tasks

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReturnPathArn</td>
<td>The ARN of the identity that is associated with the sending authorization policy that permits you to use the email address specified in the ReturnPath parameter of SendRawEmail.</td>
</tr>
</tbody>
</table>

- **Include X-headers in the email.** X-headers are custom headers that you can use in addition to standard email headers (such as the From, Reply-To, or Subject headers). Amazon SES recognizes three X-headers that you can use to specify sending authorization parameters:

  **Important**
  Do not include these X-headers in the DKIM signature, because they are removed by Amazon SES before sending the email.

<table>
<thead>
<tr>
<th>X-Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-SES-SOURCE-ARN</td>
<td>Corresponds to the SourceArn.</td>
</tr>
<tr>
<td>X-SES-FROM-ARN</td>
<td>Corresponds to the FromArn.</td>
</tr>
<tr>
<td>X-SES-RETURN-PATH-ARN</td>
<td>Corresponds to the ReturnPathArn.</td>
</tr>
</tbody>
</table>

Amazon SES removes all X-headers from the email before sending it. If you include multiple instances of an X-header, Amazon SES only uses the first instance.

The following example shows an email that includes sending authorization X-headers:

```

From: sender@example.com
To: recipient@example.com
Return-Path: feedback@example.com
Subject: subject
Content-Type: multipart/alternative;
    boundary="----=_boundary"

        ----=_boundary
Content-Type: text/plain; charset=UTF-8
Content-Transfer-Encoding: 7bit

body
        ----=_boundary
Content-Type: text/html; charset=UTF-8
Content-Transfer-Encoding: 7bit

body
        ----=_boundary--
```

### SendEmail

If you use the SendEmail operation, you can specify the cross-account identity by passing in the optional parameters below. You cannot use the X-header method when you use the SendEmail operation.
Delegate Sender Tasks

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SourceArn</td>
<td>The ARN of the identity that is associated with the sending authorization policy that permits you to send for the email address specified in the Source parameter of SendRawEmail.</td>
</tr>
<tr>
<td>ReturnPathArn</td>
<td>The ARN of the identity that is associated with the sending authorization policy that permits you to use the email address specified in the ReturnPath parameter of SendRawEmail.</td>
</tr>
</tbody>
</table>

The following example shows how to send an email that includes the SourceArn and ReturnPathArn attributes using the SendEmail operation and the SDK for Python.

```python
import boto3
from botocore.exceptions import ClientError

# Create a new SES resource and specify a region.
client = boto3.client('ses', region_name='us-west-2')

# Try to send the email.
try:
    # Provide the contents of the email.
    response = client.send_email(
        Destination={
            'ToAddresses': [
                'recipient@example.com',
            ],
        },
        Message={
            'Body': {
                'Html': {
                    'Charset': 'UTF-8',
                    'Data': 'This email was sent with Amazon SES.',
                },
            },
            'Subject': {
                'Charset': 'UTF-8',
                'Data': 'Amazon SES Test',
            },
        },
        Source='sender@example.com',
        ReturnPath='feedback@example.com'
    )
    # Display an error if something goes wrong.
    except ClientError as e:
        print(e.response['Error']['Message'])
    else:
        print("Email sent! Message ID:")
        print(response['ResponseMetadata']['RequestId'])
```

Using the Amazon SES SMTP interface

When you use the Amazon SES SMTP interface for cross-account sending, you have to include the X-SES-SOURCE-ARN, X-SES-FROM-ARN and X-SES-RETURN-PATH-ARN headers in your message. Pass these headers after you issue the DATA command in the SMTP conversation.
Using dedicated IP addresses with Amazon SES

When you create a new Amazon SES account, your emails are sent from IP addresses that are shared with other Amazon SES users. For an additional monthly charge, you can lease dedicated IP addresses that are reserved for your exclusive use. Both of these options offer unique benefits and drawbacks, which are summarized in the following table; click an item in the Benefit column for additional information about that benefit.

<table>
<thead>
<tr>
<th>Benefit</th>
<th>Shared IP addresses</th>
<th>Dedicated IP addresses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ready to use with no additional setup (p. 173)</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Reputation managed by AWS (p. 174)</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Good for customers with continuous, predictable sending patterns (p. 174)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Good for customers with less predictable sending patterns (p. 174)</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Good for high-volume senders (p. 174)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Good for low-volume senders (p. 174)</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Additional monthly costs (p. 174)</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Complete control over sender reputation (p. 174)</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Isolate reputation by email type, recipient, or other factors (p. 175)</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Provides known IP addresses that never change (p. 175)</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Important**

If you don't plan to send large volumes of email on a regular and predictable basis, we recommend that you use shared IP addresses. If you use dedicated IP addresses in situations where you are sending low volumes of mail, or if your sending patterns are highly irregular, you might experience deliverability issues.

**Ease of Setup**

If you choose to use shared IP addresses, then you don't need to perform any additional configuration. Your Amazon SES account is ready to send email as soon as you verify an email address and move out of the sandbox.

If you choose to lease dedicated IP addresses, you have to submit a request (p. 175) and optionally configure dedicated IP pools (p. 179).
Reputation Managed by AWS

IP address reputations are based largely on historical sending patterns and volume. An IP address that sends consistent volumes of email over a long period of time typically has a good reputation.

Shared IP addresses are used by several Amazon SES customers. Together, these customers send a large volume of email. AWS carefully manages this outbound traffic in order to maximize the reputations of the shared IP addresses.

If you use dedicated IP addresses, it is your responsibility to maintain your sender reputation by sending consistent and predictable volumes of email.

Note
If you would like to see Smart Network Data Services (SNDS) data for your dedicated IPs, see SNDS metrics for dedicated IPs (p. 385) for more information.

Predictability of Sending Patterns

An IP address with a consistent history of sending email has a better reputation than one that suddenly starts sending out large volumes of email with no prior sending history.

If your email sending patterns are irregular—that is, they don't follow a predictable pattern—then shared IP addresses are probably a better fit for your needs. When you use shared IP addresses, you can increase or decrease your email sending patterns as the situation demands.

If you use dedicated IP addresses, you must warm up those addresses by sending an amount of email that gradually increases every day. The process of warming up new IP addresses is described in Warming up Dedicated IP Addresses (p. 178). Once your dedicated IP addresses are warmed up, you must then maintain a consistent sending pattern.

Volume of Outbound Email

Dedicated IP addresses are best suited for customers who send large volumes of email. Most internet service providers (ISPs) only track the reputation of a given IP address if they receive a significant volume of mail from that address. For each ISP with which you want to cultivate a reputation, you should send several hundred emails within a 24-hour period at least once per month.

In some cases, you may be able to use dedicated IP addresses if you don't send large volumes of email. For example, dedicated IP addresses may work well if you send to a small, well-defined group of recipients whose mail servers accept or reject email using a list of specific IP addresses, rather than IP address reputation.

Additional Costs

The use of shared IP addresses is included in the standard Amazon SES pricing. Leasing dedicated IP addresses incurs an extra monthly cost beyond the standard costs associated with sending email using Amazon SES. Each dedicated IP address incurs a separate monthly charge. For pricing information, see the Amazon SES pricing page.

Control over Sender Reputation

When you use dedicated IP addresses, your Amazon SES account is the only one that is able to send email from those addresses. For this reason, the sender reputation of the dedicated IP addresses that you lease is determined by your email sending practices.
Ability to Isolate Sender Reputation

By using dedicated IP addresses, you can isolate your sender reputation for different components of your email program. If you lease more than one dedicated IP address for use with Amazon SES, you can create dedicated IP pools—groups of dedicated IP addresses that can be used for sending specific types of email. For example, you can create one pool of dedicated IP addresses for sending marketing email, and another for sending transactional email. To learn more, see Creating Dedicated IP Pools (p. 179).

Known, Unchanging IP Addresses

When you use dedicated IP addresses, you can find the values of the addresses that send your mail in the Dedicated IPs page of the Amazon SES console. Dedicated IP addresses don’t change.

With shared IP addresses, you don’t know the IP addresses that Amazon SES uses to send your mail, and they can change at any time.

Requesting and Relinquishing Dedicated IP Addresses

This section describes how to request and relinquish dedicated IP addresses by submitting a request in the AWS Support Center. We charge your account an additional monthly fee for each dedicated IP address that you lease for use with Amazon SES. For more information about the costs associated with dedicated IP addresses, see Amazon SES Pricing.

Best Practices for Working with Dedicated IP Addresses

Although there’s no minimum commitment, we recommend that you lease more than one dedicated IP address in each AWS Region where you use Amazon SES. Each AWS Region consists of multiple physical locations, called Availability Zones. When you lease more than one dedicated IP address, we distribute those addresses as evenly as possible across the Availability Zones in the AWS Region that you specified in your request. Distributing your dedicated IP addresses across Availability Zones in this way increases the availability and redundancy of your dedicated IP addresses.

For a list of all of the Regions where Amazon SES is currently available, see AWS Regions and Endpoints in the Amazon Web Services General Reference. To learn more about the number of Availability Zones that are available in each Region, see AWS Global Infrastructure.

Requesting Dedicated IP Addresses

The following steps show how to request dedicated IP addresses by creating a service quota increase case in the AWS Support Center. You can use this process to request as many dedicated IP addresses as you need.

To request dedicated IP addresses

2. On the Support menu, choose Support Center, as shown in the following image.
3. Choose Create case.
4. Under Create case, choose Service limit increase.
5. Under Case classification, complete the following sections:
   - For Limit type, choose SES Sending Limits.
   - For Mail Type, choose the type of email that you plan to send using your dedicated IP address. If multiple values apply, choose the option that applies to the majority of the email that you plan to send.
   - For Website URL, enter the URL of your website. Providing this information helps us better understand the type of content that you plan to send.
   - For My email sending complies with the AWS Service Terms and AUP, choose the option that applies to your use case.
   - For I only send to recipients who have specifically requested my mail, choose the option that applies to your use case.
   - For I have a process to handle bounces and complaints, choose the option that applies to your use case.
6. Under Requests, complete the following sections:
   - For Region, choose the AWS Region that your request applies to.
   - For Limit, choose Desired Maximum Send Rate.
   - For New limit value, enter the maximum number of messages that you need to be able to send per second. We use the value that you provide to calculate the number of dedicated IP addresses that you need to implement your use case. For this reason, the estimate that you provide should be as accurate as possible.

**Note**
A single dedicated IP address can only be used in the AWS Region that you chose in this step. If you want to request dedicated IP addresses for use in another AWS Region, choose Add another request, and then complete the Region, Limit, and New limit value fields for
the additional Region. Repeat this process for each Region that you want to use dedicated IP addresses in.

7. Under **Case description**, for **Use case description**, state that you want to request dedicated IP addresses. If you want to request a specific number of dedicated IP addresses, mention that as well. If you don't specify a number of dedicated IP addresses, we'll provide the number of dedicated IP addresses that are necessary to meet the sending rate requirement that you specified in the previous step.

Next, describe how you plan to use dedicated IP addresses to send email using Amazon SES. Include information about why you want to use dedicated IP addresses instead of shared IP addresses. This information helps us better understand your use case.

8. Under **Contact options**, for **Preferred contact language**, choose whether you want to receive communications for this case in **English** or **Japanese**.

9. When you finish, choose **Submit**.

After you submit the form, we evaluate your request. If we grant your request, we reply to your case in Support Center to confirm that your new dedicated IP addresses are associated with your account.

---

**Relinquish Dedicated IP Addresses**

If you no longer need dedicated IP addresses that are associated with your account, you can relinquish them by completing the following steps.

**Important**

The process of relinquishing a dedicated IP address can't be reversed. If you relinquish a dedicated IP address in the middle of a month, we prorate the monthly dedicated IP usage fee, based on the number of days that have elapsed in the current month.

**To relinquish dedicated IP addresses**

2. On the **Support** menu, choose **Support Center**.
3. On the **My support cases** tab, choose **Create case**.
4. Under **Create case**, choose **Service limit increase**.
5. Under **Case classification**, complete the following sections:
   - For **Limit type**, choose **SES Sending Limits**.
   - For **Mail Type**, choose any value.
   - For **My email sending complies with the AWS Service Terms and AUP**, choose the option that applies to your use case.
   - For **I only send to recipients who have specifically requested my mail**, choose the option that applies to your use case.
   - For **I have a process to handle bounces and complaints**, choose the option that applies to your use case.
6. Under **Requests**, complete the following sections:
   - For **Region**, choose the AWS Region that your request applies to.
     **Note**
     Dedicated IP addresses are unique to each AWS Region, so it's important to select the Region that the dedicated IP address is associated with.
   - For **Limit**, choose **Desired Maximum Send Rate**.
   - For **New limit value**, enter any number. The number that you enter here isn't important—you specify the number of dedicated IPs that you want to relinquish in the next step.
**Note**
A single dedicated IP address can only be used in a single AWS Region. If you want to relinquish dedicated IP addresses that you used in other AWS Regions, choose **Add another request**. Then complete the **Region**, **Limit**, and **New limit value** fields for the additional Region. Repeat this process for each dedicated IP address that you want to relinquish.

7. Under **Case Description**, for **Use case description**, mention that you want to relinquish existing dedicated IP addresses. If you currently lease more than one dedicated IP address, include the number of dedicated IP addresses that you want to relinquish.

8. Under **Contact options**, for **Preferred contact language**, choose whether you want to receive communications for this case in **English** or **Japanese**.

9. When you finish, choose **Submit**.

After we receive your request, we send you a message that asks you to confirm that you want to relinquish your dedicated IP addresses. After you confirm that you want to relinquish the IP addresses, we remove them from your account.

**Warming up Dedicated IP Addresses**

When determining whether to accept or reject a message, email service providers consider the reputation of the IP address that sent it. One of the factors that contributes to the reputation of an IP address is whether the address has a history of sending high-quality email. Email providers are less likely to accept mail from new IP addresses that have little or no history. Email sent from IP addresses with little or no history may end up in recipients' junk mail folders, or may be blocked altogether.

When you start sending email from a new IP address, you should gradually increase the amount of email you send from that address before using it to its full capacity. This process is called **warming up** the IP address.

The amount of time required to warm up an IP address varies between email providers. For some email providers, you can establish a positive reputation in around two weeks, while for others it may take up to six weeks. When warming up a new IP address, you should send emails to your most active users to ensure that your complaint rate remains low. You should also carefully examine your bounce messages and send less email if you receive a high number of blocking or throttling notifications. For information about monitoring your bounces, see Monitoring your Amazon SES sending activity (p. 259).

**Automatically Warm up Dedicated IP Addresses**

When you request dedicated IP addresses, Amazon SES automatically warms them up to improve the delivery of emails you send. The automatic IP address warm-up feature is enabled by default.

The steps that happen during the automatic warm-up process depend on whether or not you already have dedicated IP addresses:

- When you request dedicated IP addresses for the first time, Amazon SES distributes your email sending between your dedicated IP addresses and a set of addresses that are shared with other Amazon SES customers. Amazon SES gradually increases the number of messages sent from your dedicated IP addresses over time.

- If you already have dedicated IP addresses, Amazon SES distributes your email sending between your existing dedicated IPs (which are already warmed up) and your new dedicated IPs (which are not warmed up). Amazon SES gradually increases the number of messages sent from your new dedicated IP addresses over time.
Creating Dedicated IP Pools

**Note**
Automatic IP warm-up is a time-based process. The warm-up percentage steadily increases over 45 days independently from your sending volume.

After you warm up a dedicated IP address, you should send around 1,000 emails every day to each email provider that you want to maintain a positive reputation with. You should perform this task on each dedicated IP address that you use with Amazon SES.

You should avoid sending large volumes of email immediately after the warm-up process is complete. Instead, slowly increase the number of emails you send until you reach your target volume. If an email provider sees a large, sudden increase in the number of emails being sent from an IP address, they may block or throttle the delivery of messages from that address.

**Disable the Automatic Warm-up Process**

When you purchase new dedicated IP addresses, Amazon SES automatically warms them up for you. If you prefer to warm up dedicated IP addresses yourself, you can disable the automatic warm-up feature.

**Important**
If you disable the automatic warm up feature, you are responsible for warming up your dedicated IP addresses yourself. If you send email from addresses that haven't been warmed up, you may experience poor delivery rates.

**To disable the automatic warm-up feature**
1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation bar on the left, choose **Dedicated IPs**.
3. Clear the box next to **Automatic IP warm-up**.

**Restart the Automatic Warm-up Process**

You can restart the automatic IP warm-up process for a set of IP addresses that belong to a dedicated IP pool.

**To restart the automatic warm-up process**
1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation bar on the left, choose **Dedicated IPs**.
3. In the dedicated IP pool for which you want to restart the warm-up process, choose **Actions**, and then choose **Restart IP warm up**.

The status of the automatic warm-up process is in the **Warm Up Status** column; when the warm-up process is finished, this column will say **Complete**.

**Creating Dedicated IP Pools**

If you purchased several dedicated IP addresses to use with Amazon SES, you can create groups of those addresses. These groups are called **dedicated IP pools**. A common scenario is to create one pool of dedicated IP addresses for sending marketing communications, and another for sending transactional emails. Your sender reputation for transactional emails is then isolated from that of your marketing emails. In this scenario, if a marketing campaign generates a large number of complaints, the delivery of your transactional emails is not impacted.
This section contains procedures for creating dedicated IP pools.

Note
You can also create configuration sets that use a pool of IP addresses that are shared by all Amazon SES customers. The shared IP pool is useful in situations where you need to send email that doesn't align with your usual sending behaviors. For information about using the shared IP pool with a configuration set, see Managing dedicated IP pools (p. 256).

To create a dedicated IP pool using the Amazon SES console

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane on the left side of the screen, under Email Sending, choose Dedicated IPs.
3. Choose Create a New IP Pool.
4. On the IP Pool Name page, for Pool name, type a descriptive name for the dedicated IP pool, and then choose Next.
5. On the Add Dedicated IPs page, check the box next to each IP address you want to add to the pool, and then choose Next.

Note
Dedicated IP addresses that you haven't yet assigned to a pool are included in the ses-default-dedicated-pool. If you send an email using a configuration set that doesn't specify a sending pool, or if you send an email without specifying a configuration set at all, Amazon SES sends the email from one of the addresses in the ses-default-dedicated-pool. A dedicated IP address can only belong to one pool. If you select a dedicated IP address that's associated with a different pool, Amazon SES overwrites that setting, and associates the address with the pool that you're creating.

6. On the Assign to a configuration set page, do one of the following:
   - Select Add this pool to an existing configuration set to associate the dedicated IP pool with an existing configuration set. Then, under Existing configuration sets, choose the configuration set that you want to associate the IP pool with.
   - Select Create a new configuration set to create a configuration set and associate the dedicated IP pool with it. For Configuration set name, type a descriptive name for the configuration set.

When you finish, choose Next.
7. On the Review page, verify the settings of the dedicated IP pool. When you are ready to create the IP pool, choose Create.

Using Your Own IP Addresses to Send Email Using Amazon SES

Amazon SES includes a feature called Bring Your Own IP (BYOIP), which makes it possible to use your own IP addresses to send email through Amazon SES. If you already use a range of IP addresses to send email, you can request that we make your IP range available for sending email through Amazon SES.

BYOIP is helpful, for example, when you have developed a positive IP reputation using an in-house email sending system, but you want to migrate to Amazon SES. By using BYOIP, you can start sending email through Amazon SES immediately, without having to re-establish the reputations of your IP addresses.

Requirements

To use BYOIP, your IP address range has to meet the following requirements:
• The address range has to be registered with your Regional internet registry (RIR), such as the American Registry for Internet Numbers (ARIN), Réseaux IP Européens Network Coordination Centre (RIPE NCC), or Asia-Pacific Network Information Centre (APNIC). The address range has to be registered to a business or institutional entity and can't be registered to a person.
• You have to be able to provide proof that you own the address range by submitting a signed authorization message.
• The addresses in the IP address range have to have a clean history. We might investigate the reputation of the IP address range, and we reserve the right to reject an IP address range if it contains IP addresses that have poor reputations or are associated with malicious behavior.

Considerations
There are several factors that you should consider before you request the transfer of your IP ranges to Amazon SES:
• The most specific address range that you can specify is /24. In other words, if you transfer the IP range 203.0.113.0/24 to your Amazon SES account, then you can send from a total of 256 addresses, ranging from 203.0.113.0 to 203.0.113.255. You have to transfer the entire range—Amazon SES doesn't currently allow you to transfer individual IP addresses.
• If you use BYOIP for a specific range of IP addresses, you can only access that range from a single AWS Region.
• You can bring five address ranges per Region to your AWS account.
• If you use your own IP addresses, you can't use the addresses in the pool of shared Amazon SES IP addresses. If you need to use these shared IP addresses, you can use Amazon SES in a different AWS Region, or create a new AWS account.
• There is a monthly charge for each IP address that you use with BYOIP. For more information, see Amazon SES Pricing.

Using Your Own IP Addresses with Amazon SES
In order to prevent our systems from being used to send unsolicited or malicious content, we have to consider each BYOIP request carefully.

If you want to use your own IP range with Amazon SES please send the following information to ses-byoip-request@amazon.com:
• Your AWS account ID.
• The AWS Region that you want to use the IP range in, such as ap-south-1.
• A description of your use case.
• The IP range that you want to use with Amazon SES.
• The name of the internet registry that the range is registered with.

We'll respond to your request within 48 business hours. In our communications with you, we might request additional information, including documents that prove your ownership of the IP range.

Testing email sending in Amazon SES
Amazon SES includes a mailbox simulator that you can use to test how your application handles different email sending scenarios. The mailbox simulator is useful when, for example, you need to test an email sending application without creating fictitious email addresses, or when you need to find your system's maximum throughput without impacting your daily sending quota.
Important Considerations

Consider the following features and limitations when you use the Amazon SES mailbox simulator:

- You can use the mailbox simulator even if your account is in the Amazon SES sandbox.
- Emails that you send to the mailbox simulator are limited by your account's maximum sending rate, but they don't affect your daily sending quota. For example, if your account is authorized to send 10,000 messages per 24-hour period, and you send 100 messages to the mailbox simulator, you can still send up to 10,000 messages to regular recipients without reaching your sending quota.
- Emails that you send to the mailbox simulator don't impact your email deliverability or reputation metrics. For example, if you send a large number of messages to the bounce address of the email simulator, it doesn't cause the reputation dashboard (p. 369) to display a message warning you that your bounce rate is too high.
- For billing purposes, emails that you send to the Amazon SES mailbox simulator are the same as any other email you send using Amazon SES. In other words, we bill you the same amount for messages that you send to the mailbox simulator as for those that you send to regular recipients.
- The mailbox simulator supports labeling, which enables you to send emails to the same mailbox simulator address in multiple ways, or to see how your application handles Variable Envelope Return Path (VERP). For example, you can send an email to bounce+label1@simulator.amazonses.com and bounce+label2@simulator.amazonses.com to see if your application can match a bounce message with the email address that caused the bounce.
- If you use the mailbox simulator to simulate multiple bounces from the same sending request, Amazon SES combines the bounce responses into a single response.

Using the Mailbox Simulator

To use the email simulator, find the scenario in the following table, and then send an email to the corresponding email address.

**Note**
When you send an email to a mailbox simulator address, you must send it through Amazon SES, by using the AWS CLI, an AWS SDK, the Amazon SES console, the Amazon SES SMTP interface, or the Amazon SES API. The mailbox simulator doesn't respond to emails that it receives from external sources.

<table>
<thead>
<tr>
<th>Simulated scenario</th>
<th>Email address</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Successful delivery</strong>—The recipient's email provider accepts your email. If you set up delivery notifications as described in Monitoring Amazon SES email sending using notifications (p. 265), Amazon SES sends you a delivery notification through Amazon Simple Notification Service (Amazon SNS).</td>
<td><a href="mailto:success@simulator.amazonses.com">success@simulator.amazonses.com</a></td>
</tr>
<tr>
<td><strong>Bounce</strong>—The recipient's email provider rejects your email with an SMTP 550 5.1.1 (&quot;Unknown User&quot;) response code. Amazon SES generates a bounce notification and, depending on how you set up your account, sends it to you in an email or sends a notification to an Amazon SNS topic. The mailbox simulator email address isn't placed on the Amazon SES suppression list, which would normally happen when a hard bounce occurs. The bounce response that you receive from the</td>
<td><a href="mailto:bounce@simulator.amazonses.com">bounce@simulator.amazonses.com</a></td>
</tr>
</tbody>
</table>
Simulated scenario | Email address
--- | ---
mailbox simulator is compliant with **RFC 3464**. For information about how to receive bounce feedback, see Monitoring Amazon SES email sending using notifications (p. 265). | ooto@simulator.amazonses.com
**Automatic responses**—The recipient's email provider accepts your email and delivers it to the recipient's inbox. The email provider sends an automatic response, such as an "out of the office" (OOTO) message, to the address in the Return-Path header of the email, or the envelope sender ("MAIL FROM") address if the Return-Path header isn't present. The automatic response that you receive from the mailbox simulator is compliant with **RFC 3834**. | complaint@simulator.amazonses.com
**Complaint**—The recipient's email provider accepts your email and delivers it to the recipient's inbox. The recipient decides that your message is unsolicited and clicks "Mark as Spam" in his or her email client. Amazon SES then forwards the complaint notification to you by email or by notifying an Amazon SNS topic, depending on how you set up your account. The complaint response that you receive from the mailbox simulator is compliant with **RFC 5965**. For information about how to receive complaint feedback, see Monitoring Amazon SES email sending using notifications (p. 265). | suppressionlist@simulator.amazonses.com
**Recipient address on suppression list**—Amazon SES generates a hard bounce as if the recipient's address is on the global suppression list. |

Testing Reject Events

Every message that you send through Amazon SES is scanned for viruses. If you send a message that contains a virus, Amazon SES accepts the message, detects the virus, and rejects the entire message. When Amazon SES rejects the message, it stops processing the message, and doesn't attempt to deliver it to the recipient's mail server. It then generates a Reject event.

The Amazon SES mailbox simulator doesn't include an address for testing Reject events. However, you can test Reject events by using an EICAR test file. This file is an industry-standard method of testing antivirus software in a safe manner. To create an EICAR test file, paste the following text into a file:

```
X50!P%@AP\4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
```

Save the file as `sample.txt`, attach it to an email, and then send the email to a verified address. If there are no other issues with the email, Amazon SES accepts the message, but then rejects it as it would if it contained an actual virus.

**Note**

Rejected emails—including those that you send by using the procedure above—count against your daily sending quota. We bill you for each message that you send, including rejected messages.
Using the account-level suppression list

Amazon SES includes an account-level suppression list that applies to your AWS account in the current AWS Region. This suppression list prevents you from sending email to addresses that previously produced a bounce or complaint event. When you configure the account-level suppression list, you specify whether addresses should be added to the list when they result in hard bounces, when they result in complaints, or both. You can manually add or remove individual or bulk addresses from the account-level suppression list by using the Amazon SES API v2.

Note
To bulk add or remove addresses, you must have production access. To learn more about the sandbox, see Moving out of the Amazon SES sandbox (p. 71).

Amazon SES also includes a global suppression list. For more information, see Using the Amazon SES global suppression list (p. 192).

Account-level suppression list considerations

You should consider the following factors when you use the account-level suppression list:

- If you started using Amazon SES after November 25, 2019, your account uses the account-level suppression list by default for both bounces and complaints. If you started using Amazon SES before this date, then you have to enable this feature by using the PutAccountSuppressionAttributes operation in the Amazon SES API.
- If you attempt to send a message to an address that's on the account-level suppression list, Amazon SES accepts the message, but doesn't send it.
- Amazon SES doesn't count the messages that you send to addresses on the account-level suppression list toward the bounce rate for your account.
- Amazon SES counts the messages that you send to addresses on the account-level suppression list toward your daily sending quota.
- Email addresses on the account-level suppression list remain there until you remove them by using the DeleteSuppressedDestination operation in the Amazon SES API v2.
- If your account's ability to send email is paused, Amazon SES automatically deletes the addresses in the account-level suppression list after 90 days. If your account's ability to send email is restored before this 90-day period ends, then the addresses in the account-level suppression list aren't deleted.
- Gmail doesn't provide complaint data to Amazon SES. If a recipient uses the Spam button in the Gmail web client to report a message that they receive from you as spam, they aren't added to the account-level suppression list.
- You can enable the account-level suppression list if your account is in the Amazon SES sandbox. However, you can't use the PutSuppressedDestination or CreateImportJob operation until your account is removed from the sandbox. To learn more about the sandbox, see Moving out of the Amazon SES sandbox (p. 71).
- When you use the account-level suppression list, Amazon SES also adds addresses that result in hard bounces to the global suppression list.

Enabling the account-level suppression list

You can use the PutAccountSuppressionAttributes operation in the Amazon SES API v2 to enable and set up the account-level suppression list. You can quickly and easily configure this setting by using the AWS...
Enabling the account-level suppression list for a configuration set

You can also configure the account-level suppression so that it only applies to specific configuration sets (p. 249). When you do, addresses are only added to the suppression list if you specified the configuration set when you sent the email that caused the bounce or complaint event.

Note

The following procedure assumes that you’ve already installed the AWS CLI. For more information about installing and configuring the AWS CLI, see the AWS Command Line Interface User Guide.

To configure the account-level suppression list for a configuration set by using the AWS CLI

At the command line, enter the following command:

```
Linux, macOS, or Unix
aws sesv2 put-configuration-set-suppression-options
  --configuration-set-name configSet
  --suppressed-reasons BOUNCE COMPLAINT

Windows
aws sesv2 put-configuration-set-suppression-options
  --configuration-set-name configSet
  --suppressed-reasons BOUNCE COMPLAINT
```

In the preceding example, replace `configSet` with the name of the configuration set that should use the account-level suppression list.
Manually adding individual email addresses to the account-level suppression list

You can manually add individual addresses to the account-level suppression list by using the `PutSuppressedDestination` operation in the Amazon SES API v2. There's no limit to the number of addresses that you can add to the account-level suppression list.

**Note**
The following procedure assumes that you've already installed the AWS CLI. For more information about installing and configuring the AWS CLI, see the [AWS Command Line Interface User Guide](#).

**To manually add an address to the account-level suppression list by using the AWS CLI**

- At the command line, enter the following command:

  Linux, macOS, or Unix
  ```bash
  aws sesv2 put-suppressed-destination --email-address recipient@example.com --reason BOUNCE
  ```

  Windows
  ```bash
  aws sesv2 put-suppressed-destination --email-address recipient@example.com --reason BOUNCE
  ```

  In the preceding example, replace `recipient@example.com` with the email address that you want to add to the account-level suppression list, and `BOUNCE` with the reason that you're adding the address to the suppression list (acceptable values are `BOUNCE` and `COMPLAINT`).

Adding email addresses in bulk to the account-level suppression list

You can manually add addresses in bulk by first uploading your contact list into an Amazon S3 object that you have permission to access followed by using the `CreateImportJob` operation in the Amazon SES API v2.

**Note**
There's no limit to the number of addresses that you can add to the account-level suppression list, but there is a bulk add limit of 1,000,000 addresses in an Amazon S3 object per API call.

To add email addresses in bulk to your account-level suppression list, complete the following steps.

- Upload your address list into an Amazon S3 object in either CSV or JSON format.

  CSV format example for adding addresses:
  ```
  recipient1@example.com, BOUNCE
  recipient2@example.com, COMPLAINT
  ```

  Only newline-delimited JSON files are supported. In this format, each line is a complete JSON object that contains an individual address definition.
Viewing a list of addresses that are on the account-level suppression list

JSON format example for adding addresses:

```json
{"emailAddress":"recipient1@example.com","reason":"BOUNCE"}

{"emailAddress":"recipient2@example.com","reason":"COMPLAINT"}
```

In the preceding examples, replace `recipient1@example.com` and `recipient2@example.com` with the email addresses that you want to add to the account-level suppression list. The acceptable reasons that you're adding the addresses to the suppression list are `BOUNCE` and `COMPLAINT`.

- Use the `CreateImportJob` operation in the Amazon SES API v2.

**Note**
The following procedure assumes that you've already installed the AWS CLI. For more information about installing and configuring the AWS CLI, see the AWS Command Line Interface User Guide.

To manually add bulk addresses to the account-level suppression list by using the AWS CLI

- At the command line, enter the following command:

  **Linux, macOS, or Unix**

  ```bash
  aws sesv2 create-import-job \
  --import-destination "{"SuppressionListDestination":"\n  {"SuppressionListImportAction":"PUT"}}" \
  --import-data-source "{"S3Url": "s3://s3bucket/s3object"},{"DataFormat": "CSV"}
  ```

  **Windows**

  ```bash
  aws sesv2 create-import-job \
  --import-destination "{"SuppressionListDestination":"\n  {"SuppressionListImportAction":"PUT"}}" \
  --import-data-source "{"S3Url": "s3://s3bucket/s3object"},{"DataFormat": "CSV"}
  ```

  In the preceding examples, replace `s3bucket` and `s3object` with the Amazon S3 bucket name and Amazon S3 object name.

Viewing a list of addresses that are on the account-level suppression list

You can view a list of all of the email addresses that are on the account-level suppression list for your account by using the `ListSuppressedDestinations` operation in the Amazon SES API v2.

**Note**
The following procedure assumes that you've already installed the AWS CLI. For more information about installing and configuring the AWS CLI, see the AWS Command Line Interface User Guide.

To view a list of all of the email addresses that are on the account-level suppression list

- At the command line, enter the following command:
Amazon Simple Email Service Classic Developer Guide

Viewing a list of addresses that are on the account-level suppression list

```
aws sesv2 list-suppressed-destinations
```

The preceding command returns all of the email addresses that are in the account-level suppression list for your account. The output resembles the following example:

```
{
    "SuppressedDestinationSummaries": [
        {
            "EmailAddress": "recipient2@example.com",
            "Reason": "COMPLAINT",
            "LastUpdateTime": 1586552585.077
        },
        {
            "EmailAddress": "recipient0@example.com",
            "Reason": "COMPLAINT",
            "LastUpdateTime": 1586552666.613
        },
        {
            "EmailAddress": "recipient1@example.com",
            "Reason": "BOUNCE",
            "LastUpdateTime": 1586556479.141
        }
    ]
}
```

You can use the `StartDate` option to only show email addresses that were added to the list after a certain date.

**To view a list of addresses that were added to the account-level suppression list after a specific date**

- At the command line, enter the following command:

  ```
  aws sesv2 list-suppressed-destinations --start-date 1604394130
  ```

  In the preceding command, replace `1604394130` with the Unix timestamp of the start date.

You can also use the `EndDate` option to only show email addresses that were added to the list before a certain date.

**To view a list of addresses that were added to the account-level suppression list before a specific date**

- At the command line, enter the following command:

  ```
  aws sesv2 list-suppressed-destinations --end-date 1611126000
  ```

  In the preceding command, replace `1611126000` with the Unix timestamp of the end date.

On the Linux, macOS, or Unix command line, you can also use the built-in `grep` utility to search for specific addresses or domains.

**To search the account-level suppression list for a specific address**

- At the command line, enter the following command:
Removing an email address from the account-level suppression list

If an address is on the suppression list for your account, but you know that the address shouldn't be on the list, you can manually remove it by using `DeleteSuppressedDestination` operation in the Amazon SES API v2.

**Note**
The following procedure assumes that you've already installed the AWS CLI. For more information about installing and configuring the AWS CLI, see the AWS Command Line Interface User Guide.

**To remove an address from the account-level suppression list by using the AWS CLI**

- At the command line, enter the following command:
  
  **Linux, macOS, or Unix**
  ```bash
  aws sesv2 delete-suppressed-destination \
  --email-address recipient@example.com
  ```

  **Windows**
  ```powershell
  aws sesv2 delete-suppressed-destination `\n  --email-address recipient@example.com
  ```

  In the preceding example, replace `recipient@example.com` with the email address that you want to remove from the account-level suppression list.

Removing email addresses in bulk from the account-level suppression list

You can manually remove addresses in bulk by first uploading your contact list into an Amazon S3 object that you have permission to access followed by using the `CreateImportJob` operation in the Amazon SES API v2.

**Note**
There's no limit to the number of addresses that you can remove from the account-level suppression list, but there is a bulk delete limit of 10,000 addresses in an Amazon S3 object per API call.

To remove email addresses in bulk from your account-level suppression list, complete the following steps.

- Upload your address list into an Amazon S3 object in either CSV or JSON format.
  
  CSV format example for removing addresses:
recipient3@example.com

Only newline-delimited JSON files are supported. In this format, each line is a complete JSON object that contains an individual address definition.

JSON format example for adding addresses:

{"emailAddress":"recipient3@example.com"}

In the preceding examples, replace recipient3@example.com with the email addresses that you want to remove from the account-level suppression list.

- Use the CreateImportJob operation in the Amazon SES API v2.

**Note**
The following procedure assumes that you've already installed the AWS CLI. For more information about installing and configuring the AWS CLI, see the AWS Command Line Interface User Guide.

**To manually remove bulk addresses from the account-level suppression list by using the AWS CLI**

- At the command line, enter the following command:

  **Linux, macOS, or Unix**

  ```bash
  aws sesv2 create-import-job
  --import-destination "{"SuppressionListDestination":"{"SuppressionListImportAction":"DELETE"}"}"
  --import-data-source "{"S3Url": "s3://s3bucket/s3object"},{"DataFormat": "CSV"}"
  ```

  **Windows**

  ```bash
  aws sesv2 create-import-job ` `--import-destination "{"SuppressionListDestination":"{"SuppressionListImportAction":"DELETE"}"} ` `--import-data-source "{"S3Url": "s3://s3bucket/s3object"},{"DataFormat": "CSV"} ` ```

  In the preceding examples, replace s3bucket and s3object with the Amazon S3 bucket name and Amazon S3 object name.

**Viewing a list of import jobs for the account**

You can view a list of all of the email addresses that are on the account-level suppression list for your account by using the ListImportJobs operation in the Amazon SES API v2.

**Note**
The following procedure assumes that you've already installed the AWS CLI. For more information about installing and configuring the AWS CLI, see the AWS Command Line Interface User Guide.

**To view a list of all of the import jobs for the account**

- At the command line, enter the following command:
Getting information about an import job for the account

You can get information about an import job for the account by using the `GetImportJob` operation in the Amazon SES API v2.

**Note**
The following procedure assumes that you've already installed the AWS CLI. For more information about installing and configuring the AWS CLI, see the [AWS Command Line Interface User Guide](https://docs.aws.amazon.com/cli/latest/userguide/getting-started.html).

**To get information about an import job for the account**

- At the command line, enter the following command:

```
aws sesv2 get-import-job --job-id JobId
```
The preceding command returns information about an import job for the account. The output resembles the following example:

```json
{
   "ImportDataSource": {
      "S3Url": "s3://bucket/object",
      "DataFormat": "CSV"
   },
   "ProcessedRecordsCount": 2,
   "FailureInfo": {
      "FailedRecordsS3Url": "s3presignedurl"
   },
   "JobStatus": "COMPLETED",
   "JobId": "jobid",
   "CreatedTimestamp": 1597251915.243,
   "FailedRecordsCount": 1,
   "ImportDestination": {
      "SuppressionListDestination": {
         "SuppressionListImportAction": "PUT"
      }
   },
   "CompletedTimestamp": 1597252002.583
}
```

Disabling the account-level suppression list

You can use the `PutAccountSuppressionAttributes` operation in the Amazon SES API v2 to effectively disable the account-level suppression list by removing the values from the `suppressed-reasons` attribute.

**Note**
The following procedure assumes that you've already installed the AWS CLI. For more information about installing and configuring the AWS CLI, see the AWS Command Line Interface User Guide.

To disable the account-level suppression list by using the AWS CLI

- At the command line, enter the following command:

  ```bash
  aws sesv2 put-account-suppression-attributes --suppressed-reasons
  ```

Using the Amazon SES global suppression list

Amazon SES includes a *global suppression list*. When any Amazon SES customer sends an email that results in a hard bounce, Amazon SES adds the email address that produced the bounce to a global suppression list. The global suppression list is *global* in the sense that it applies to all Amazon SES customers. In other words, if a different customer attempts to send an email to an address that's on the global suppression list, Amazon SES accepts the message, but doesn't send it, because the email address is suppressed. An advantage of the global suppression list is that it applies to all Amazon SES accounts by default—you don't have to perform any additional configuration to use it. A disadvantage is that you can't query the global suppression list, because it contains email addresses that are associated with other Amazon SES users' accounts. Also, you can't manually add addresses to the global suppression list, and you can only remove addresses from the global suppression list by using the Amazon SES console.

Amazon SES also includes an account-level suppression list. For more information, see Using the account-level suppression list (p. 184).
Global suppression list considerations

You should consider the following factors when you use the global suppression list:

- The global suppression list is enabled by default for all Amazon SES accounts. You can't disable it.
- Because Amazon SES applies the global suppression list to all customers, you can't query the global suppression list or add addresses to it manually.
- When an email address produces a hard bounce, Amazon SES adds the address to the global suppression list for a short period of time. After that period of time elapses, Amazon SES removes the address from the list. If the address produces another hard bounce, Amazon SES adds it back to the global suppression list for a longer period of time, and removes it at the end of that period. The amount of time that an address remains on the global suppression list increases each time the address produces a hard bounce. An address can remain on the global suppression list for up to 14 days.
- If you attempt to send a message to an address that's on the global suppression list, Amazon SES accepts the message, but doesn't send it. Amazon SES generates a bounce notification with a bounceType value of Permanent, and a bounceSubType value of Suppressed. Receiving this type of bounce notification is the only way to know if an address is on the global suppression list. You can't query the global suppression list.
- Amazon SES counts the messages that you send to addresses on the global suppression list toward the bounce rate for your account.
- Amazon SES counts the messages that you send to addresses on the global suppression list toward your daily sending quota.
- As with any email address that produces a hard bounce, you should remove addresses that cause a suppression list bounce from your mailing list unless you're certain that the address is valid. Suppression list bounces count towards your account's bounce rate. If your bounce rate gets too high, we might place your account under review or pause your account's ability to send email.

Removing an address from the global suppression list

If you're sure that an address on the global suppression list is actually a valid recipient, you can remove it by using the following procedure. When you remove an address from the global suppression list in one Region, the removal applies to all AWS accounts in all regions.

To remove an email address from the global suppression list

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. Use the Region selector in the top right corner of the window to choose one of the following AWS Regions: US East (N. Virginia), US West (Oregon), or Europe (Ireland). The specific Region that you choose isn't important, because the suppression list applies equally to all Regions.
   
   Note
   To complete this procedure, you have to use one of the AWS Regions listed in the preceding paragraph. You can't complete this procedure in other AWS Regions.
3. In the navigation pane, choose Suppression List Removal.
4. In the Email Address field, type the email address that you want to remove from the global suppression list.
5. In the Type characters field, type the characters that you see in the image above it.
6. Choose Submit.

If the email address that you specify is on the global suppression list, we remove it immediately.
Using list management

Amazon SES offers list management capabilities, which means customers can manage their own mailing lists, known as contact lists. A contact list is a list that allows you to store all of your contacts that have subscribed to a particular topic or topics. A contact is an end-user who is receiving your emails. A topic is an interest group, theme, or label within a list. Lists can have multiple topics.

By using the ListContacts operation in the Amazon SES API v2, you can retrieve a list of all your contacts who have subscribed to a particular topic, to whom you can send emails using the SendEmail operation.

You can manually add or remove individual or bulk addresses from the account-level suppression list by using the Amazon SES API v2.

Note
To bulk add or remove addresses, you must have production access. To learn more about the sandbox, see Moving out of the Amazon SES sandbox (p. 71).

For information about subscription management, see Using subscription management (p. 199).

List management overview

You should consider the following factors when you use list management:

- You can specify list topics while creating the list.
- Only one contact list is allowed per AWS account.
- A list can have a maximum of 20 topics.
- You can update an existing contact list, including adding new topics to the list, adding or deleting contacts from a list, and updating contact preferences for a list or topic.
- You can update topic metadata, such as the topic display name or description.
- You can get a list of contacts in a contact list, contacts subscribed to a topic, contacts unsubscribed from a topic, and contacts unsubscribed from all topics in the list.
- You can import your existing contact lists to Amazon SES using the CreateImportJob API.
- Amazon SES will bounce an email if it is sent to an unsubscribed contact on your contact list. For more information, see Using subscription management (p. 199).
- Each contact can have associated attributes which you can use to store information about that contact.

Configuring list management

You can use the following operations to configure list management capabilities. For the full list of contact list and contact operations, see the Amazon SES API v2 Reference.

Create a contact list

You can use the CreateContactList operation in the Amazon SES API v2 to create a contact list. You can quickly and easily configure this setting by using the AWS CLI. For more information about installing and configuring the AWS CLI, see the AWS Command Line Interface User Guide.

To create a contact list by using the AWS CLI

- At the command line, enter the following command:

```
aws sesv2 create-contact-list --cli-input-json file://CONTACT-LIST-JSON
```
In the preceding command, replace `CONTACT-LIST-JSON` with the path to your JSON file for your `CreateContactList` request.

An example `CreateContactList` input JSON file for the request is as follows:

```json
{
    "ContactListName": "ExampleContactListName",
    "Description": "Creating a contact list example",
    "Topics": [
        {
            "TopicName": "Sports",
            "DisplayName": "Sports Newsletter",
            "Description": "Sign up for our free newsletter to receive updates on all sports.",
            "DefaultSubscriptionStatus": "OPT_OUT"
        },
        {
            "TopicName": "Cycling",
            "DisplayName": "Cycling newsletter",
            "Description": "Never miss a cycling update by subscribing to our newsletter.",
            "DefaultSubscriptionStatus": "OPT_IN"
        },
        {
            "TopicName": "NewProducts",
            "DisplayName": "New products",
            "Description": "Hear about new products by subscribing to this mailing list.",
            "DefaultSubscriptionStatus": "OPT_IN"
        },
        {
            "TopicName": "DailyUpdates",
            "DisplayName": "Daily updates",
            "Description": "Start your day with sport updates, Monday through Friday.",
            "DefaultSubscriptionStatus": "OPT_OUT"
        }
    ]
}
```

**Create a contact**

You can use the `CreateContact` operation in the Amazon SES API v2 to create a contact. You can quickly and easily configure this setting by using the AWS CLI. For more information about installing and configuring the AWS CLI, see the AWS Command Line Interface User Guide.

**To create a contact by using the AWS CLI**

- At the command line, enter the following command:

  ```bash
  aws sesv2 create-contact --cli-input-json file://CONTACT-JSON
  ```

  In the preceding command, replace `CONTACT-JSON` with the path to your JSON file for your `CreateContact` request.

  An example `CreateContact` input JSON file for the request is as follows:

  ```json
  {
    "ContactListName": "ExampleContactListName",
    "EmailAddress": "example@amazon.com",
    "UnsubscribeAll": false,
  }
  ```
In the example above, an UnsubscribeAll value of false shows that the contact has not unsubscribed from all topics, where a value of true would mean the contact has unsubscribed from all topics.

TopicPreferences includes information about the contact’s subscription status to topics. In the preceding example, the contact has opted in to the "Sports" topic and will receive all emails to the "Sports" topic.

The AttributesData is a JSON field where you can put any metadata about our contact. It must be a valid JSON object.

**Bulk importing contacts to your contact list**

You can manually add addresses in bulk by first uploading your contacts into an Amazon S3 object that you have permission to access followed by using the CreateImportJob operation in the Amazon SES API v2.

You should create a contact list before importing your contacts.

**Note**
You can add up to 1 million contacts to a contact list per ImportJob.

To add contacts in bulk to your contact list, complete the following steps.

- Upload your contacts into an Amazon S3 object in either CSV or JSON format.

  **CSV format**

  The first line of the file that is uploaded to Amazon S3 should be a header line.

  The topicPreferences object needs to be flattened for the CSV format. Every topic in the topicPreferences will have a separate header field.

  CSV format example for adding contacts in bulk to a contact list:

  ```
  emailAddress,unsubscribeAll,attributesData,topicPreferences.Sports,topicPreferences.Cycling
  example1@amazon.com,false,{"Name": "John"},OPT_IN,OPT_OUT
  example2@amazon.com,true,,OPT_OUT,OPT_OUT
  ```

  **JSON format**

  Only newline-delimited JSON files are supported. In this format, each line is a complete JSON object that contains one contact's information.

  JSON format example for adding contacts in bulk to a contact list:

  ```
  {
    "emailAddress": "example1@amazon.com",
  }
  ```
In the preceding examples, replace example1@amazon.com and example2@amazon.com with the email addresses you want to add to the contact list. Replace the attributesData values with the values specific to the contact. Additionally, replace Sports and Cycling with the topicName that applies to your contact. The acceptable topicPreferences are OPT_IN and OPT_OUT.

The following attributes are supported when uploading your contacts into an Amazon S3 object in either CSV or JSON format:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>emailAddress</td>
<td>The contact's email address. This is a mandatory field.</td>
</tr>
<tr>
<td>unsubscribeAll</td>
<td>A boolean value status noting if the contact is unsubscribed from all contact list topics.</td>
</tr>
<tr>
<td>topicPreferences</td>
<td>The contact's preferences for being opted-in to or opted-out of topics.</td>
</tr>
<tr>
<td>attributesData</td>
<td>The attribute data attached to a contact.</td>
</tr>
</tbody>
</table>

• Use the CreateImportJob operation in the Amazon SES API v2.

Note
The following procedure assumes that you’ve already installed the AWS CLI. For more information about installing and configuring the AWS CLI, see the AWS Command Line Interface User Guide.
To manually add contacts in bulk to your contact list by using the AWS CLI

- At the command line, enter the following command:

  **Linux, macOS, or Unix**

  ```
  aws sesv2 create-import-job
  --import-destination "{""ContactListDestination": {""ContactListName": "ExampleContactListName", "ContactListImportAction": "PUT"}}"
  --import-data-source "{""S3Url": "s3://s3bucket/s3object",""DataFormat": "CSV"}"
  ```

  **Windows**

  ```
  aws sesv2 create-import-job
  --import-destination "{""ContactListDestination": {""ContactListName": "ExampleContactListName", "ContactListImportAction": "PUT"}}"
  --import-data-source "{""S3Url": "s3://s3bucket/s3object",""DataFormat": "CSV"}"
  ```

  In the preceding examples, replace `s3bucket` and `s3object` with the Amazon S3 bucket name and Amazon S3 object name.

**List your contacts to send them emails**

You can use the `ListContacts` operation to retrieve a list of all your contacts who have subscribed to a particular topic, in conjunction with the `SendEmail` operation, which allows you to send them emails.

**To list contacts by using the AWS CLI**

1. At the command line, enter the following command:

  ```
  aws sesv2 list-contacts --cli-input-json file://LIST-CONTACTS-JSON
  ```

  In the preceding command, replace `LIST-CONTACTS-JSON` with the path to your JSON file for your `ListContacts` request.

  An example `ListContacts` input JSON file for the request is as follows:

  ```
  {
  "ContactListName": "ExampleContactListName",
  "Filter": {
  "FilteredStatus": "OPT_IN",
  "TopicFilter": {
  "TopicName": "Cycling",
  "UseDefaultIfPreferenceUnavailable": true
  },
  "PageSize": 50
  }
  ```

  The `FilteredStatus` shows the subscription status for which you want to filter, which is either `OPT_IN` or `OPT_OUT`.

  The `TopicFilter` is an optional filter which specifies which topic you want results for, and in the example above, that is "Cycling."
UseDefaultIfPreferenceUnavailable can have a value of true or false. If true, the topic default preference will be used if the contact doesn't have any explicit preference for a topic. If false, only contacts with an explicitly set preference are considered for filtering.

2. After listing the contacts in your list using the above ListContacts operation, use the SendEmail operation to send emails to each of your contacts. When using the SendEmail operation, you can specify ListManagementOptions to enable Amazon SES to add unsubscribe links to your email. For more information, see Using subscription management (p. 199).

If you include ListManagementOptions in your SendEmail request to a recipient email address that is not on your contact list, then a contact will be created on your list automatically.

Amazon SES will bounce an email if it is sent to an unsubscribed contact on your contact list, which means you won’t need to update your SendEmail requests to avoid sending to contacts who have unsubscribed.

To use the SendEmail operation, include the contactListName and topicName to which the email belongs, as follows. The topicName is optional.

```plaintext
ListManagementOptions:
  String contactListName
  String topicName
```

3. Alternatively, you can use the X-SES-LIST-MANAGEMENT-OPTIONS header to specify a list and topic name while sending email using SMTP interface.

To specify a list and topic name while sending email using the SMTP interface, add the following email header to your message:

```
X-SES-LIST-MANAGEMENT-OPTIONS: {contactListName}; topic={topicName}
```

Using subscription management

Amazon SES provides a subscription management capability, in which Amazon SES automatically enables the unsubscribe links in every outgoing email when you specify the contactListName and topicName within ListManagementOptions in the SendEmail operation request.

If a contact unsubscribes from a particular topic or list, Amazon SES does not allow email sending to the contact for that topic or list in the future.

**Note**
Subscription management is available for those using Easy DKIM in Amazon SES (p. 131), but it’s not possible for Amazon SES to add the unsubscribe links to your email for senders who are signing emails themselves before calling Amazon SES.

For information about list management, see Using list management (p. 194).

Subscription management overview

You should consider the following factors when you use subscription management:

- Subscription management will be fully managed by Amazon SES. This means that Amazon SES receives unsubscribe emails and requests from the unsubscribe webpage and then updates the contact’s preference in your list. You can receive unsubscribe notifications using configuration set notifications. For more information about configuration sets, see Using Amazon SES configuration sets (p. 249).
• You need to specify the contact list while sending the email. Subscription management via the Link-Unsubscribe header and footer links will be handled accordingly.
• Amazon SES adds support for the List-Unsubscribe header standards, which will enable email clients and inbox providers to display an unsubscribe link at the top of the email if they support it.
• List-Unsubscribe headers follow the following behavior:
  If a contact clicks the List-Unsubscribe header link in an email which has both the contact list and topic specified, then the contact will be unsubscribed only from that specific topic.
  If the topic is not specified, then the contact will be unsubscribed from all the topics in the list.
• Contacts will be taken to an unsubscribe landing page when they click an unsubscribe link in the email footer.
• The unsubscribe landing page will give contacts an option to update their preferences, meaning OPT_IN or OPT_OUT, for all the topics in a particular list. The landing page also gives an option to unsubscribe from all topics in the list.
• You must include a placeholder in your emails to indicate where Amazon SES needs to insert the unsubscribe URL. You can include the placeholder two times maximum. If used more than two times, only the first two occurrences are replaced.
• The List-Unsubscribe header and footer link are added only if the email is being sent to a single recipient, and the placeholder must present in order to add the unsubscribe link.
• For transactional emails where you don't want contacts to be able to unsubscribe, you can omit the ListManagementOptions field with your SendEmail request.

Adding an unsubscribe footer link

You will need to use the {{amazonSESUnsubscribeUrl}} placeholder in templated and non-templated emails to specify where Amazon SES needs to insert the unsubscribe URL.

Placeholder replacement is supported only for HTML and TEXT content types.

You can include the placeholder two times maximum. If used more than two times, only the first two occurrences are replaced.

Unsubscribe header considerations

Subscription management through an unsubscribe header is enabled by the the following two headers, which Amazon SES adds. If you're adding these headers in your own email before sending them to Amazon SES, Amazon SES overrides them if you have enabled ListManagementOptions.

List-Unsubscribe
List-Unsubscribe-Post

Contacts who unsubscribe by clicking the link provided by the unsubscribe header will have a different experience depending on their email client or inbox provider. Users who unsubscribe via the link in the header will not have the option of choosing which topics they unsubscribe from, and they will be unsubscribed from the topic to which the email was sent.

For more information about message headers, see RFC 2369 and RFC 8058.

Amazon SES and security protocols

This topic describes the security protocols that you can use when you connect to Amazon SES, as well as when Amazon SES delivers an email to a receiver.
Email Sender to Amazon SES

The security protocol that you use to connect to Amazon SES depends on whether you are using the Amazon SES API or the Amazon SES SMTP interface, as described next.

HTTPS

If you are using the Amazon SES API (either directly or through an AWS SDK), then all communications are encrypted by TLS through the Amazon SES HTTPS endpoint. The Amazon SES HTTPS endpoint supports TLS 1.2, TLS 1.1, and TLS 1.0.

SMTP Interface

If you are accessing Amazon SES through the SMTP interface, you are required to encrypt your connection using Transport Layer Security (TLS). Note that TLS is often referred to by the name of its predecessor protocol, Secure Sockets Layer (SSL).

Amazon SES supports two mechanisms for establishing a TLS-encrypted connection: STARTTLS and TLS Wrapper.

- **STARTTLS**—STARTTLS is a means of upgrading an unencrypted connection to an encrypted connection. There are versions of STARTTLS for a variety of protocols; the SMTP version is defined in RFC 3207. For STARTTLS connections, Amazon SES supports TLS 1.2, TLS 1.1, TLS 1.0 and SSLv2Hello.

- **TLS Wrapper**—TLS Wrapper (also known as SMTPS or the Handshake Protocol) is a means of initiating an encrypted connection without first establishing an unencrypted connection. With TLS Wrapper, the Amazon SES SMTP endpoint does not perform TLS negotiation: it is the client's responsibility to connect to the endpoint using TLS, and to continue using TLS for the entire conversation. TLS Wrapper is an older protocol, but many clients still support it. For TLS Wrapper connections, Amazon SES supports TLS 1.2, TLS 1.1 and TLS 1.0.

For information about connecting to the Amazon SES SMTP interface using these methods, see Connecting to an Amazon SES SMTP endpoint (p. 90).

Amazon SES to Receiver

Amazon SES supports TLS 1.2, TLS 1.1 and TLS 1.0 for TLS connections.

By default, Amazon SES uses opportunistic TLS. This means that Amazon SES always attempts to make a secure connection to the receiving mail server. If Amazon SES can't establish a secure connection, it sends the message unencrypted.

You can change this behavior by using configuration sets. Use the PutConfigurationSetDeliveryOptions API operation to set the TlsPolicy property for a configuration set to Require. You can use the AWS CLI to make this change.

To configure Amazon SES to require TLS connections for a configuration set

- At the command line, enter the following command:

  ```bash
  ```

  In the preceding example, replace *MyConfigurationSet* with the name of your configuration set.
When you send an email using this configuration set, Amazon SES only sends the message to the receiving email server if it can establish a secure connection. If Amazon SES can’t make a secure connection to the receiving email server, it drops the message.

**End-to-End Encryption**

You can use Amazon SES to send messages that are encrypted using S/MIME or PGP. Messages that use these protocols are encrypted by the sender. Their contents can only be viewed by recipients who possess the public keys that are required to decrypt the messages.

Amazon SES supports the following MIME types, which you can use to send S/MIME encrypted email:

- application/pkcs7-mime
- application/pkcs7-signature
- application/x-pkcs7-mime
- application/x-pkcs7-signature

Amazon SES also supports the following MIME types, which you can use to send PGP-encrypted email:

- application/pgp-encrypted
- application/pgp-keys
- application/pgp-signature
Amazon Simple Email Service Classic Developer Guide
Email receiving concepts

Receiving email with Amazon SES

Amazon Simple Email Service (Amazon SES) is a mail server that can both send and receive mail on your behalf. When you use Amazon SES to receive your mail, Amazon SES handles underlying mail-receiving operations, such as:

- communicating with other mail servers
- scanning for spam and viruses
- rejecting mail from untrusted sources
- accepting mail for recipients in your domain

When you receive email, Amazon SES processes it according to instructions you provide. For example, Amazon SES can deliver incoming mail to an Amazon S3 bucket, publish it to an Amazon SNS topic, or send it to Amazon WorkMail. You can also create rules that explicitly block or allow all messages from specific IP address ranges, or that automatically send bounce messages when messages are sent to specific email addresses.

**Note**

Amazon SES doesn't include POP or IMAP servers for receiving incoming email. This means that you can't use an email client such as Microsoft Outlook to receive incoming email. If you need a solution that can both send and receive email by using an email client, consider using Amazon WorkMail.

Amazon SES only supports email receiving in certain AWS Regions. For a complete list of Regions where email receiving is supported, see Amazon Simple Email Service endpoints and quotas in the AWS General Reference.

**Topics in this section:**

- Amazon SES email receiving concepts (p. 203)
- Getting started receiving email with Amazon SES (p. 205)
- Setting up Amazon SES email receiving (p. 210)
- Managing email receiving in Amazon SES (p. 231)

Amazon SES email receiving concepts

When you use Amazon SES as your email receiver, you must tell the service what to do with your mail. The primary method, which gives you fine-grained control over your mail, is to specify the actions to take based on the recipient. The other method is to block or allow mail based on the originating IP address. This topic describes both methods.

**Recipient-based control**

The primary way to control your incoming mail is to specify how mail is handled based on its recipient. For example, if you own `example.com`, you can specify that mail for `user@example.com` should bounce, and that all other mail for `example.com` and its subdomains should be delivered. The list of recipients you provide is called the **condition**.

You set up **receipt rules** to specify how to handle the mail when a condition is satisfied. A receipt rule consists of a condition and an ordered list of actions. If the recipient to whom the incoming mail is addressed matches a recipient specified in the condition, then Amazon SES performs the actions specified in the rule. The following actions are available:
• **S3 action**—Delivers the mail to an Amazon S3 bucket and, optionally, notifies you through Amazon SNS.

• **SNS action**—Publishes the mail to an Amazon SNS topic.

  **Note**
  The SNS action includes a complete copy of the email content in the Amazon SNS notifications. The other Amazon SNS notifications mentioned here simply notify you of email delivery; they contain information about the email, not the email content itself.

• **Lambda action**—Calls your code through a Lambda function and, optionally, notifies you through Amazon SNS.

• **Bounce action**—Rejects the email by returning a bounce response to the sender and, optionally, notifies you through Amazon SNS.

• **Stop action**—Terminates the evaluation of the receipt rule set and, optionally, notifies you through Amazon SNS.

• **Add header action**—Adds a header to the received email. You typically use this action only in combination with other actions.

• **WorkMail action**—Handles the mail with Amazon WorkMail. You will typically not use this action directly because Amazon WorkMail takes care of the setup.

Receipt rules are grouped together into _receipt rule sets_. You can define multiple receipt rule sets for your AWS account, but only one receipt rule set is active at any time. The following figure shows how receipt rules, receipt rule sets, and actions relate to each other.

---

**IP address-based control**

You can control your mail flow on a broader level by setting up _IP address filters_. IP address filters are optional and enable you to specify whether to accept or reject mail originating from an IP address or range of IP addresses. Your IP address filters can include _block lists_ (IP addresses from which you want to block incoming mail) and _allow lists_ (IP addresses from which you want to always accept mail). IP address filters are useful for blocking spam. Amazon SES maintains its own block list of IP addresses known to send spam, but you can choose to receive mail from those IP addresses by adding them to your allow list.

**Note**
If you want to allow mail that originates from an Amazon EC2 IP address, you must add it to your allow list. All mail originating from Amazon EC2 is blocked by default.

**Email-receiving process**

When Amazon SES receives an email for your domain, the following events occur:
1. Amazon SES first looks at the IP address of the sender. Amazon SES allows the mail to pass this stage unless:
   - The IP address is in your block list.
   - The IP address is in the Amazon SES block list and not on your allow list.
2. Amazon SES examines your active receipt rule set to determine whether any of your receipt rules contain a condition that matches any of the incoming email's recipients.
3. If there aren't any matches, Amazon SES rejects the mail. Otherwise, Amazon SES accepts the mail.
4. If Amazon SES accepts the mail, it evaluates your active receipt rule set. All of the receipt rules that match at least one of the recipient conditions are applied in the order that they are defined, unless an action or a receipt rule explicitly terminates evaluation of the receipt rule set.

Now that you have an overview of the process, you can get started by going to Setting up email receiving (p. 210).

Getting started receiving email with Amazon SES

In this tutorial, you'll create an AWS account, register a domain using Amazon Route 53, and configure Amazon Simple Email Service to deliver all email sent to your domain to an Amazon Simple Storage Service bucket.

Note
Amazon SES only supports email receiving in certain AWS Regions. For a complete list of Regions where email receiving is supported, see Amazon Simple Email Service Endpoints and Quotas in the AWS General Reference.

Topics
- Step 1: Before you begin (p. 205)
- Step 2: Verify your domain (p. 206)
- Step 3: Set up a receipt rule (p. 207)
- Step 4: Send a test email (p. 209)
- Step 5: View the received email (p. 209)
- Step 6: Clean up (p. 210)

Step 1: Before you begin

Before you start this tutorial, sign up for an AWS account (if you don't already have one), and use Amazon Route 53 to register the domain you want to use to receive email.

Note
Amazon SES only supports email receiving in certain AWS Regions. For a complete list of Regions where email receiving is supported, see Amazon Simple Email Service Endpoints and Quotas in the AWS General Reference.

Sign up for AWS

If you already have an AWS account, you can skip this section.

To create an AWS account

1. Go to https://console.aws.amazon.com/ses/, and then choose Get Started with Amazon SES.
2. On the Create an AWS Account page, complete the required fields and follow the on-screen instructions to create a new account.
Register a domain using Route 53

This tutorial assumes that you’re using a domain that you registered using Route 53. You can also use a domain that you registered using another service, but the procedures for verifying your domain will differ from those shown in this tutorial. For more information about using Route 53 to register a domain, see Register a New Domain in the Amazon Route 53 Developer Guide.

You can also transfer an existing domain to Route 53. For more information about transferring domains to Route 53, see Transferring Registration for a Domain to Route 53 in the Amazon Route 53 Developer Guide.

Next step: Step 2: Verify your domain (p. 206)

Step 2: Verify your domain

Before you can configure Amazon SES to receive email for your domain, you must prove that you own the domain. You can verify any domain that you own, but it is easier to verify domains that you registered using Route 53.

Note
If your account is still in the Amazon SES sandbox, complete the procedure in Moving out of the Amazon SES sandbox (p. 71) before you complete the procedure in this section.

To verify a domain with Amazon SES

1. Open the Amazon SES console at https://console.aws.amazon.com/ses/.

   Note
   To complete the procedure in this section, sign in to the AWS Management Console using the same AWS account you used when you registered your domain with Route 53.

2. In the navigation pane, under Identity Management, choose Domains.

3. Choose Verify a New Domain.

4. On the Verify a New Domain dialog box, for Domain, type the name of the domain that you registered using Route 53, and then choose Verify This Domain.

5. On the Verify a New Domain dialog box, choose Use Route 53.

   Note
   If you don’t see the Use Route 53 button, your domain may not be registered with Route 53. If you used another service to register your domain, you can verify the domain by completing the procedures in Verifying a domain with Amazon SES (p. 60).

6. On the Use Route 53 dialog box, select Domain Verification Record and Email Receiving Record. Then, under Hosted Zones, select the name of the Hosted Zone you want to use. If you haven’t made any changes to the domain you registered using Route 53, there should only be one option available in the Hosted Zones section.

   Important
   If you’ve already set up mail exchanger (MX) records for your domain, the next step will replace those records with new ones.

7. Choose Create Record Sets. You’ll return to the list of domains.

8. Wait five minutes, and then choose the refresh (ícia) button. Confirm that the value in the Status column is verified. If the status is pending verification, wait a few more minutes, and then refresh the list again. Repeat this process until the domain’s status is verified.

Next step: Step 3: Set up a receipt rule (p. 207)
Step 3: Set up a receipt rule

To use Amazon SES as your email receiver, you must have an active receipt rule set. A receipt rule set is a collection of receipt rules that specify what Amazon SES should do with mail it receives for your verified domains. Because you're setting up email receiving with Amazon SES for the first time, Amazon SES automatically creates a default receipt rule set for you. The receipt rule you create in this section belongs to the default receipt rule set.

Note
The procedures in this section assume you've never created a receipt rule set. If your account already contains a receipt rule set, you'll need to make the receipt rule you create in this section active before Amazon SES applies it to the incoming email for your domain. For more information about enabling and disabling receipt rule sets, see Activating and disabling a receipt rule set (p. 232).

To create a receipt rule

1. In the navigation pane, under Email Receiving, choose Rule Sets.
2. Choose Create a Receipt Rule.
3. On the Recipients page, choose Next Step.

   Note
   Because you aren't adding any recipients, Amazon SES applies this rule to all recipients across all of your verified domains.

4. For Add action, choose S3.

5. For S3 bucket, choose Create S3 bucket.
6. For **Bucket Name**, type a name for the Amazon S3 bucket. The bucket name you enter must meet the following requirements:

- It can only contain lowercase letters, numbers, periods (.), and hyphens (-).
- It must be unique across all of AWS.
- It must start and end with a number or a lowercase letter.
- It must contain at least 3 characters, and no more than 63 characters.
- It can't be formatted as an IP address (for example, 192.168.5.4).
- It can't contain two adjacent periods (..) or a dash adjacent to a period (-. or .-).

When you finish, choose **Create Bucket**.

**Note**
Because you're using the Amazon SES console to create an Amazon S3 bucket, Amazon SES automatically creates and applies a policy that gives it permission to write to the bucket. However, if you choose an existing Amazon S3 bucket, you must give Amazon SES permission to write to the bucket by attaching a policy to the bucket (p. 215) using the Amazon S3 console or API.

7. Choose **Next Step**.

8. On the **Rule Details** page, for **Rule name**, type `my-rule`. Select the check box next to **Enabled**, and then choose **Next Step**.
9. On the **Review** page, choose **Create Rule**.

**Next step: Step 4: Send a test email (p. 209)**

**Step 4: Send a test email**

Now that you've verified and configured your domain, you can send an email to test your domain's ability to receive email.

To send a test email, use an email account that you know is capable of sending email, such as your personal email address. Send a test message to any email address on your verified domain. For example, if your domain is `example.com`, you can send an email to `test@example.com` or `abc123@example.com` (or any other address on the `example.com` domain).

**Note**
You don't need to complete any additional steps to create individual email addresses on your domain—Amazon SES receives every email that is sent to any address on the verified domain, and applies the receipt rule you created in **Step 3: Set up a receipt rule (p. 207)**.

**Next step: Step 5: View the received email (p. 209)**

**Step 5: View the received email**

After you send a test message to an address on your domain, you can retrieve it from your Amazon S3 bucket and view its contents.

**To view a message that you received through Amazon SES**

1. Open the Amazon S3 console at [https://console.aws.amazon.com/s3/](https://console.aws.amazon.com/s3/).
2. In the Amazon S3 console, choose the bucket you created in **Step 3: Set up a receipt rule (p. 207)**.
3. In the Amazon S3 bucket, find the email you received. The name of the email is a unique string of letters and numbers.
Note
The bucket may also contain a file named AMAZON_SES_SETUP_NOTIFICATION. You can ignore or delete this file.

4. Select the check box next to the name of the file. On the Actions menu, choose Download.
5. Open the folder on your computer that contains the file you downloaded in the preceding step. There are several ways to view the downloaded message, including the following:
   • Open the file in a text editor and read its contents directly. Depending on the method you used to send the email, part of the message may be encoded. If part of the message is encoded, you'll need to decode them manually (for example, by using a base64 decoder).
   • Add the .eml extension to the end of the file name, and then open the file using an email client such as Microsoft Outlook or Mozilla Thunderbird. Most email clients will automatically decode the encoded parts of a message, and will display things like HTML formatting and file attachments.

Next step: Step 6: Clean up (p. 210)

Step 6: Clean up

After you complete this tutorial, you can clean up the resources you created to avoid incurring additional charges.

Amazon SES receipt rule set

If you no longer want Amazon SES to receive mail for your domain, you can disable the active receipt rule set (p. 232).

Amazon S3 bucket

If you no longer want the Amazon S3 bucket that you created, you can delete it. To delete a bucket, you must first delete its contents. For more information about deleting folders and buckets, see Delete an Object and Bucket in the Amazon Simple Storage Service Getting Started Guide.

Route 53 domain

If you no longer want to use Route 53 to register your domain, you can delete the registration or transfer the domain to another registrar.

Setting up Amazon SES email receiving

This section describes what you need to do to configure Amazon SES to receive your mail. For example, you should first consider how you want to receive, filter, and process your mail, because those decisions will affect how you configure Amazon SES. You also need to verify your domain with Amazon SES to prove that you own it, and then configure your domain to send incoming mail to Amazon SES. Next, you give Amazon SES permission to access AWS resources. Finally, you configure email receiving by creating a receipt rule set, receipt rules, and IP address filters.

These steps are explained in the following topics:
   • Considering your use case for Amazon SES email receiving (p. 211)
   • Verifying your domain for Amazon SES email receiving (p. 213)
Considering your use case for Amazon SES email receiving

Before you set up Amazon SES to receive your mail, you might find it helpful to consider the following questions.

General considerations

Is your Amazon SES account still in the sandbox?

If you haven't already done so, complete the procedure in Moving out of the Amazon SES sandbox (p. 71) to have your account removed from the Amazon SES sandbox.

Email content

How do you want Amazon SES to pass you the email content?

Amazon SES can provide you the email content in two ways: it can store the emails in an Amazon S3 bucket that you specify, or it can send you an Amazon SNS notification that contains a copy of the email. Amazon SES delivers you the raw, unmodified email in Multipurpose Internet Mail Extensions (MIME) format. For more information about MIME format, see RFC 2045.

How large are the emails that you'll be receiving?

If you store emails in an Amazon S3 bucket, the maximum email size (including headers) is 30 MB. If you receive your emails through Amazon SNS notifications, the maximum email size (including headers) is 150 KB.

How do you want to trigger the processing of your mail?

After your mail is delivered, you will want to process it with your own code. For example, your application might convert the base 64-encoded email into a displayable format and then make it available to an end user through an email client. There are a couple of ways you can start the process:

- If your emails are delivered to Amazon S3, your application can listen for Amazon SNS notifications generated by S3 actions, extract the message ID of the email from the notifications, and then use the message ID to retrieve the email from Amazon S3.

Alternatively, you can incorporate email processing into your receipt rules by writing a Lambda function. In this case, your receipt rule should first write the email to Amazon S3, and then trigger the Lambda function. Lambda actions can be executed synchronously or asynchronously from within your receipt rules, depending on whether the Lambda function needs to return a result that influences how other actions are executed. We recommend that you use asynchronous execution unless synchronous is absolutely necessary for your use case. For more information about AWS Lambda, see the AWS Lambda Developer Guide.
• If your emails are delivered through an Amazon SNS notification by using the SNS action, your application can listen for Amazon SNS notifications, and then extract the email messages from the notifications.

Do you want the emails to be encrypted?

Amazon SES integrates with AWS Key Management Service (AWS KMS) to optionally encrypt the mail it writes to your Amazon S3 bucket. Amazon SES uses client-side encryption to encrypt your mail before writing it to Amazon S3. This means that you must decrypt the content on your side after retrieving the mail from Amazon S3. The AWS SDK for Java and AWS SDK for Ruby provide a client that can handle the decryption for you. Amazon SES can encrypt the emails for you only if you choose for your emails to be delivered to an Amazon S3 bucket.

Unwanted mail

At what point in the email-receiving process do you want to reject unwanted mail?

When a sender tries to send an email to a recipient, the sender's email server exchanges a sequence of commands with the recipient's server. This sequence is called the SMTP conversation.

You can reject incoming email at two points in the email receiving process: during the SMTP conversation, and after the SMTP conversation. You use IP address filters to reject messages during the SMTP conversation, and receipt rules to reject emails after the SMTP conversation.

You can use IP address filters to reject email that originates from specific IP addresses. The benefit of using IP address filters to reject unwanted mail is that we don't charge you for messages that are rejected during the SMTP conversation. The drawback to using IP address filters is that they reject email from the IP addresses you specify without performing any analysis on the actual content of the messages. For more information about IP address filters, see Creating IP address filters for Amazon SES email receiving (p. 218).

You can use receipt rules to send a bounce notification to the sender of an email based on the address (or domain, or subdomain) that the message was sent to. The benefit of using receipt rules is that you can perform additional analysis on incoming messages before you send a bounce notification to the sender. For example, you can use AWS Lambda to send bounce notifications only when messages fail DKIM authentication or are identified as spam. The drawback to using receipt rules is that, because receipt rules are processed after the SMTP conversation, we bill you for each message that you receive. You might also be charged if you use Lambda to analyze the content of incoming messages. For more information about receipt rules, see Creating receipt rules for Amazon SES email receiving (p. 219). For more information about using Lambda to analyze incoming email, see Lambda function examples (p. 227).

Using other AWS services

Have you set up the appropriate permissions?

If you want your mail to be delivered to an Amazon S3 bucket, published to an Amazon SNS topic you don't own, trigger a Lambda function, or use a custom master AWS KMS key, you need to give Amazon SES permission to access those resources. To give Amazon SES access, you create policies on resources from the consoles or APIs for those AWS services. For more information Giving permission (p. 215).

Mail streams

How do you want to divide your mail stream?

Your domain most likely receives different classes of mail. For example, some of your domain's mail, such as an email to user@example.com, might be intended for a personal inbox. Other mail, such as an
email to `unsubscribe@example.com`, might be better directed to automated systems instead. You can use receipt rules to divide your incoming mail so that it can be processed differently. For information about how to set up receipt rules, see Creating receipt rules (p. 219).

Regional availability

Does Amazon SES support email receiving in your Region?

Amazon SES only supports email receiving in certain AWS Regions. For a complete list of Regions where email receiving is supported, see Amazon Simple Email Service endpoints and quotas in the AWS General Reference.

Verifying your domain for Amazon SES email receiving

As with any domain you want to use for sending or receiving email with Amazon SES, you must first prove that you own it. The verification procedure, which includes initiating domain verification with Amazon SES and then publishing a TXT record to your DNS server, is described in Verifying domains in Amazon SES (p. 59).

Note

Although Amazon SES enables you to verify single email addresses, you must verify a domain if you want to use Amazon SES for email receiving.

You can also start the domain verification process when you set up receipt rules in Creating receipt rules (p. 219). The recipient list will indicate which recipients are not verified, and enable you to initiate verification. In any case, you must complete domain verification by publishing a TXT record to your DNS server, as described in Amazon SES domain verification TXT records (p. 63).

You can confirm that your email address or domain is verified by looking at its status in the Email Address Identities or Domain Identities list in the Amazon SES console or by using the Amazon SES GetIdentityVerificationAttributes API.

Publishing an MX record for Amazon SES email receiving

A mail exchanger record (MX record) is a configuration that specifies which mail servers can accept email that's sent to your domain.

To have Amazon SES manage your incoming email, you need to add an MX record to your domain's DNS configuration. The MX record that you create refers to the endpoint that receives email for the AWS Region where you use Amazon SES. For example, the endpoint for the US West (Oregon) Region is `inbound-smtp.us-west-2.amazonaws.com`. For a complete list of endpoints, see Amazon SES regions and endpoints (p. 457).

Note

The endpoints that receive email in Amazon SES aren't IMAP or POP3 email servers. You can't use these URLs as incoming mail servers in email clients.
If you need a solution that can both send and receive email by using an email client, consider using Amazon WorkMail.

The following procedure includes general steps for creating an MX record. The specific procedures for creating an MX record depend on your DNS or hosting provider. See your provider's documentation for information about adding an MX record to the DNS configuration for your domain.
To add an MX record to the DNS configuration for your domain

1. Sign in to the management console for your DNS provider.
2. Create a new MX record.
3. For the MX record **Name**, enter your domain, followed by a period. For example, if you want Amazon SES to manage email that's sent to the domain `example.com`, enter the following:

   example.com.

   **Note**
   Some DNS providers refer to the **Name** field as the **Host**, **Domain**, or **Mail Domain**.

4. For **Type**, choose **MX**.

   **Note**
   Some DNS providers refer to the **Type** field as the **Record Type** or a similar name.

5. For **Value**, enter the following:

   10 inbound-smtp.regionInboundUrl.amazonaws.com

   In the preceding example, replace `regionInboundUrl` with the address of the endpoint that receives email for the AWS Region you use with Amazon SES. For example, if you're using the US East (N. Virginia) Region, replace `region` with `us-east-1`. For a complete list of email receiving endpoints, see Amazon SES regions and endpoints (p. 457).

   **Note**
   The management consoles of some DNS providers include separate fields for the record **Value** and the record **Priority**. If this is the case for your DNS provider, enter 10 for the **Priority** value, and enter the incoming mail endpoint URL for the **Value**.

### Instructions for creating MX records for various providers

The procedures for creating an MX record for your domain depend on which DNS provider you use. This section includes links to the documentation for several common DNS providers. This list isn't a complete list of providers. If your provider isn't listed below, you can probably still use it with Amazon SES. Inclusion on this list isn't an endorsement or recommendation of any company's products or services.

<table>
<thead>
<tr>
<th>DNS/Hosting Provider Name</th>
<th>Documentation Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Route 53</td>
<td>Creating Records by Using the Amazon Route 53 Console</td>
</tr>
<tr>
<td>GoDaddy</td>
<td>Add an MX record</td>
</tr>
<tr>
<td>DreamHost</td>
<td>How do I change my MX records?</td>
</tr>
<tr>
<td>Cloudflare</td>
<td>How do I add or edit mail or MX records?</td>
</tr>
<tr>
<td>HostGator</td>
<td>Changing MX records - Windows</td>
</tr>
<tr>
<td>Namecheap</td>
<td>How can I set up MX records required for mail service?</td>
</tr>
</tbody>
</table>
Giving permissions to Amazon SES for email receiving

Some of the tasks that you can perform when you receive email in Amazon SES, such as sending email to an Amazon S3 bucket or calling a Lambda function, require special permissions. This section includes example policies for several common use cases.

Topics in this section:
- Give Amazon SES permission to write to an Amazon S3 bucket (p. 215)
- Give Amazon SES permission to use your AWS KMS key (p. 216)
- Give Amazon SES permission to invoke a Lambda function (p. 216)
- Give Amazon SES permission to publish to an Amazon SNS topic that belongs to a different AWS account (p. 217)

Give Amazon SES permission to write to an Amazon S3 bucket

When you apply the following policy to an Amazon S3 bucket, it gives Amazon SES permission to write to that bucket. For more information about creating receipt rules that transfer incoming email to Amazon S3, see S3 action (p. 229).

For more information about attaching policies to Amazon S3 buckets, see Using Bucket Policies and User Policies in the Amazon Simple Storage Service Developer Guide.

```
{
    "Version":"2012-10-17",
    "Statement":[
        {
            "Sid":"AllowSESPuts",
            "Effect":"Allow",
            "Principal":{
                "Service":"ses.amazonaws.com"
            },
            "Action":"s3:PutObject",
            "Resource":"arn:aws:s3:::myBucket/*",
            "Condition":{
                "StringEquals":{
                    "AWS:SourceAccount": "111122223333",
                }
            }
        }
    ]
}
```

Make the following changes to the preceding policy example:

- If you’re located in a special partition such as China or AWS GovCloud (US), replace `aws` with the corresponding AWS partition, see Amazon Resource Names (ARNs) in the AWS General Reference.
- Replace `myBucket` with the name of the S3 bucket that you want to write to.
• Replace `us-west-2` with the AWS Region where you created the receipt rule.
• Replace `111122223333` with your AWS account ID.
• Replace `rule_set_name` with the name of the rule set that contains the receipt rule that contains the deliver to Amazon S3 bucket action.
• Replace `receipt_rule_name` with the name of the receipt rule that contains the deliver to Amazon S3 bucket action.

**Give Amazon SES permission to use your AWS KMS key**

In order for Amazon SES to encrypt your emails, it must have permission to use the AWS KMS key that you specified when you set up your receipt rule. You can either use the default KMS key (`aws/ses`) in your account, or use a customer managed key that you create. If you use the default KMS key, you don’t need to perform any additional steps to give Amazon SES permission to use it. If you use a customer managed key, you need to give Amazon SES permission to use it by adding a statement to the key’s policy.

Use the following policy statement as the key policy to allow Amazon SES to use your customer managed key when it receives email on your domain.

```
{
  "Sid": "AllowSESToEncryptMessagesBelongingToThisAccount",
  "Effect": "Allow",
  "Principal": { "Service":"ses.amazonaws.com" },
  "Action": [ "kms:GenerateDataKey*" ],
  "Resource": "*",
  "Condition":{
    "StringEquals":{
      "AWS:SourceAccount": "111122223333",
    }
  }
}
```

Make the following changes to the preceding policy example:

• If you’re located in a special partition such as China or AWS GovCloud (US), replace `aws` with the corresponding AWS partition, see Amazon Resource Names (ARNs) in the AWS General Reference.
• Replace `us-west-2` with the AWS Region where you created the receipt rule.
• Replace `111122223333` with your AWS account ID.
• Replace `rule_set_name` with the name of the rule set that contains the receipt rule that you’ve associated with email receiving.
• Replace `receipt_rule_name` with the name of the receipt rule that you’ve associated with email receiving.

For more information about attaching policies to AWS KMS keys, see Using Key Policies in AWS KMS in the AWS Key Management Service Developer Guide.

**Give Amazon SES permission to invoke a Lambda function**

To enable Amazon SES to call a Lambda function, you can choose the function when you create a receipt rule in the Amazon SES console. When you do, Amazon SES automatically adds the necessary permissions to the function.
Alternatively, you can use the \texttt{AddPermission} operation in the AWS Lambda API to attach a policy to a function. The following call to the \texttt{AddPermission} API gives Amazon SES permission to invoke your Lambda function. For more information about attaching policies to Lambda functions, see \texttt{AWS Lambda Permissions} in the \texttt{AWS Lambda Developer Guide}.

\begin{verbatim}
{
  "Action": "lambda:InvokeFunction",
  "Principal": "ses.amazonaws.com",
  "SourceAccount": "111122223333",
  "StatementId": "GiveSESPermissionToInvokeFunction"
}
\end{verbatim}

Make the following changes to the preceding policy example:

\begin{itemize}
  \item If you're located in a special partition such as China or AWS GovCloud (US), replace \texttt{aws} with the corresponding AWS partition, see Amazon Resource Names (ARNs) in the \texttt{AWS General Reference}.
  \item Replace \texttt{us-west-2} with the AWS Region where you created the receipt rule.
  \item Replace \texttt{111122223333} with your AWS account ID.
  \item Replace \texttt{rule_set_name} with the name of the rule set that contains the receipt rule where you created your Lambda function.
  \item Replace \texttt{receipt_rule_name} with the name of the receipt rule containing your Lambda function.
\end{itemize}

**Give Amazon SES permission to publish to an Amazon SNS topic that belongs to a different AWS account**

If you want to publish notifications to a topic in a separate AWS account, then you have to attach a policy to the Amazon SNS topic.

The following policy gives Amazon SES permission to publish to an Amazon SNS topic in a separate AWS account.

\begin{verbatim}
{
  "Version":"2008-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Principal":{
        "Service":"ses.amazonaws.com"
      },
      "Action":"SNS:Publish",
      "Condition":{
        "StringEquals":{
          "AWS:SourceAccount":"aws_account_id",
        }
      }
    }
  ]
}
\end{verbatim}

Make the following changes to the preceding policy example:

\begin{itemize}
  \item If you're located in a special partition such as China or AWS GovCloud (US), replace \texttt{aws} with the corresponding AWS partition, see Amazon Resource Names (ARNs) in the \texttt{AWS General Reference}.
\end{itemize}
Creating IP address filters for Amazon SES email receiving

An IP address filter enables you to optionally specify whether to accept or reject mail originating from an IP address or range of IP addresses.

You can use the Amazon SES console or the CreateReceiptFilter API to create an IP address filter.

Note
If you only want to receive mail from a finite list of known IP addresses, then set up a block list that contains 0.0.0.0/0, and set up an allow list that contains the IP addresses that you trust. This configuration blocks all IP addresses by default, and only allows mail from the IP addresses that you explicitly specify.

To create an IP address filter (console)

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under Email Receiving, choose IP Address Filters.
3. In the content pane, choose Create Filter.
4. For Filter Name, type a name for the IP address filter. The name must contain less than 64 alphanumeric, hyphen (-), underscore (_), and period (.) characters. The name must start and end with a letter or number.
5. For IP Address Range, type a single IP address or a range of IP addresses that you want to block or allow, specified in Classless Inter-Domain Routing (CIDR) notation. An example of a single IP address is 10.0.0.1. An example of a range of IP addresses is 10.0.0.1/24. For more information about CIDR notation, see RFC 2317.
6. For Policy Type, choose Allow or Block.
7. Choose Create Filter.

For information about how to use the CreateReceiptFilter API to create an IP address filter, see the Amazon Simple Email Service API Reference.

Creating a receipt rule set for Amazon SES email receiving

A receipt rule set is a collection of receipt rules that specify what Amazon SES should do with mail it receives across all of your domains. To use Amazon SES as your email receiver, you must create a receipt rule set for your account. For more information about the role of receipt rule sets in the email-receiving process, see Email receiving concepts (p. 203).
Creating receipt rules

Only one receipt rule set can be active at a time. However, you can create multiple receipt rule sets. For example, it may be useful to have multiple receipt rule sets if you want to maintain a record of the receipt rules you used in the past, or if you need to change receipt rules rapidly for testing purposes.

**Note**
If you do not want to use Amazon SES as your email receiver, simply disable all of your receipt rule sets. For information about how to disable receipt rule sets, see Managing receipt rule sets (p. 232).

You can use the Amazon SES console or API to create a receipt rule set.

- **Using the Amazon SES console**
  - Receipt rules exist in receipt rule sets only, so to create a receipt rule set, you can start by creating a receipt rule. For more information, see Creating receipt rules (p. 219). When you reach the end of this procedure, you can create a new receipt rule set.
  - Copy an existing receipt rule set as explained in Managing receipt rule sets (p. 232).
  - In the left navigation pane, under Email Receiving, choose Rule Sets, and then choose Create a New Rule Set.
- **Using the Amazon SES API**—Use the CreateReceiptRuleSet API to create an empty receipt rule set, as described in the Amazon Simple Email Service API Reference. Then, you can use the Amazon SES console or the CreateReceiptRule API to add receipt rules to it.

Creating receipt rules for Amazon SES email receiving

Receipt rules let you specify what Amazon SES does with email it receives for the email addresses or domains you own. A receipt rule contains a condition and an ordered list of actions. If the recipient of an incoming email matches a recipient specified in the conditions for the receipt rule, then Amazon SES performs the actions specified in that receipt rule. For more information about the role of receipt rules in the email-receiving process, see Email receiving concepts (p. 203).

**Important**
To set up receipt rules, first verify a domain and publish an MX record on that domain. For more information about verifying domains, see Verifying domains in Amazon SES (p. 59). For more information about publishing MX records, see the section called “Publishing an MX record” (p. 213).

You can use the Amazon SES console or the CreateReceiptRule API operation to create receipt rules. This section provides procedures for creating a new receipt rule using the console. These procedures assume that your Amazon SES account does not contain any existing receipt rules.

Setting up a receipt rule

You can use the Amazon SES console or the CreateReceiptRule API to create rules.

**To create a receipt rule using the console**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under Email Receiving, choose Rule Sets.
3. Choose Create a Receipt Rule.
4. Use the following procedure to add one or more recipients. Collectively, these recipients are the condition. You can have a maximum of 100 recipients per receipt rule.
   a. Under Recipients, specify the incoming email address or domain for which you want to set up a receipt rule. The following table uses the address user@example.com to show how to specify recipients.
<table>
<thead>
<tr>
<th>If you want to...</th>
<th>Specify the following recipient...</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Match a specific email address.</td>
<td><a href="mailto:user@example.com">user@example.com</a></td>
<td>Also matches variations of the address that contain labels (such as user <a href="mailto:+123@example.com">+123@example.com</a> and <a href="mailto:user+xyz@example.com">user+xyz@example.com</a>). However, if you specify an address that contains a label, only that specific address is matched.</td>
</tr>
<tr>
<td>Match all addresses within a domain, but not those within its subdomains.</td>
<td>example.com</td>
<td></td>
</tr>
<tr>
<td>Match all addresses within a specific subdomain, but not those within the parent domain.</td>
<td>subdomain.example.com</td>
<td></td>
</tr>
<tr>
<td>Match all addresses within all subdomains, but not those within the parent domain.</td>
<td>.example.com</td>
<td>Note the period (.) before the domain name.</td>
</tr>
<tr>
<td>Match all addresses within a domain, and all addresses within all of its subdomains.</td>
<td>example.com .example.com</td>
<td>Create two separate recipients: one with the domain name, and one with a period followed by the domain name.</td>
</tr>
<tr>
<td>Match all recipients in all verified domains</td>
<td>[None]</td>
<td>Leave the recipient field blank.</td>
</tr>
</tbody>
</table>

**Important**

If multiple Amazon SES accounts receive email on a common domain (for example, if multiple teams in the same company each have separate Amazon SES accounts), Amazon SES processes all matching receipt rules simultaneously for each of those accounts. This behavior may result in a situation where one account generates a bounce, while another account accepts the email.

We recommend that you coordinate with other teams in your organization that use Amazon SES to ensure that each account uses unique receipt rules, and that those rules do not overlap. In these situations, it is best to configure your receipt rules to use only email addresses or subdomains that are unique to your group or team.

b. Choose **Add Recipient**.

c. Repeat steps a and b for each recipient you want to add. When you finish adding recipients, choose **Next Step**.

5. **Use the following procedure to add one or more actions to the receipt rule.**

a. Choose an action from the menu.

b. Choose the action settings. For information about the options for each action, see Action options (p. 221).
c. Add additional actions as needed, and then choose **Next Step**.

6. For **Rule Details**, use the following procedure to choose settings.

   a. For **Rule Name**, type a name for the receipt rule. The name must contain less than 64 alphanumeric, hyphen (-), underscore (_), and period (.) characters. The name must start and end with a letter or number.

   b. If you want to enable the receipt rule, leave the **Enabled** option selected.

   c. If you want Amazon SES to reject any incoming emails that are not sent over a connection that is encrypted with Transport Layer Security (TLS), select **TLS**.

   d. If you want Amazon SES to scan incoming emails for spam and viruses, select **Enable Spam and Virus Scanning**.

7. For **Rule Set**, choose an existing receipt rule set or click **Create New Rule Set**.

8. For **Rule Position**, choose where to place the receipt rule in the ordered list of receipt rules. The receipt rules are evaluated sequentially.

9. Choose **Next Step**, and then choose **Create Rule**.

For information about how to use the `CreateReceiptRule` API to create rules, see the Amazon Simple Email Service API Reference.

**Action options**

Each receipt rule for Amazon SES email receiving contains an ordered list of actions. The overall setup procedure for receipt rules is described in Creating receipt rules for Amazon SES email receiving (p. 219). This section describes the specific options for each action type.

The action types are the following:

- **Add header action** (p. 221)
- **Bounce action** (p. 221)
- **Lambda action** (p. 222)
- **S3 action** (p. 229)
- **SNS action** (p. 230)
- **Stop action** (p. 231)
- **WorkMail action** (p. 231)

**Add header action**

The **Add Header** action adds a custom header to the received email. You typically use this action only in combination with another action. This action has the following options.

- **Header name**—The name of the header to add. It must be between 1 and 50 characters, inclusive, and consist of alphanumeric (a-z, A-Z, 0-9) characters and dashes only.

- **Header value**—The value of the header to add. It must be less than 2048 characters, and must not contain newline characters ("\r" or "\n").

**Bounce action**

The **Bounce** action rejects the email by returning a bounce response to the sender and, optionally, notifies you through Amazon SNS. This action has the following options.

- **SMTP Reply Code**—The SMTP reply code, as defined by RFC 5321.
Creating receipt rules

- **SMTP Status Code**—The SMTP enhanced status code, as defined by RFC 3463.
- **Message**—Human-readable text to include in the bounce email.
- **Reply Sender**—The email address of the sender of the bounced email. This is the address from which the bounce email will be sent. It must be verified with Amazon SES.
- **SNS Topic**—The name or ARN of the Amazon SNS topic to optionally notify when a bounce email is sent. An example of an Amazon SNS topic ARN is `arn:aws:sns:us-west-2:123456789012:MyTopic`. You can also create an Amazon SNS topic when you set up your action by choosing `Create SNS Topic`. For more information about Amazon SNS topics, see the Amazon Simple Notification Service Developer Guide.

  **Note**
  The Amazon SNS topic you choose must be in the same AWS region as the Amazon SES endpoint you use to receive email.

You can type in your own values for these fields, or you can choose a template that fills in the SMTP Reply Code, SMTP Status Code, and Message fields with values based on the bounce reason. The following templates are available:

- **Mailbox Does Not Exist**—SMTP Reply Code = 550, SMTP Status Code = 5.1.1
- **Message Too Large**—SMTP Reply Code = 552, SMTP Status Code = 5.3.4
- **Mailbox Full**—SMTP Reply Code = 552, SMTP Status Code = 5.2.2
- **Message Content Rejected**—SMTP Reply Code = 500, SMTP Status Code = 5.6.1
- **Unknown Failure**—SMTP Reply Code = 554, SMTP Status Code = 5.0.0
- **Temporary Failure**—SMTP Reply Code = 450, SMTP Status Code = 4.0.0

For additional bounce codes that you might use by typing custom values in the fields, see RFC 3463.

**Lambda action**

The Lambda action calls your code through a Lambda function and, optionally, notifies you through Amazon SNS. This action has the following options.

- **Lambda function**—The ARN of the Lambda function. An example of a Lambda function ARN is `arn:aws:lambda:us-west-2:account-id:function:MyFunction`. For information about AWS Lambda, see the AWS Lambda Developer Guide.
- **Invocation type**—The invocation type of the Lambda function. An invocation type of `RequestResponse` means that the execution of the function will immediately result in a response, and a value of `Event` means that the function will be invoked asynchronously. We recommend that you use `Event` invocation type unless synchronous execution is absolutely necessary for your use case.

  **Note**
  There is a 30-second timeout on `RequestResponse` invocations.

For information about AWS Lambda invocation types, see the AWS Lambda Developer Guide.

- **SNS Topic**—The name or ARN of the Amazon SNS topic to notify when the specified Lambda function is triggered. An example of an Amazon SNS topic ARN is `arn:aws:sns:us-west-2:123456789012:MyTopic`. You can also create an Amazon SNS topic when you set up your action by choosing `Create SNS Topic`. For more information about Amazon SNS topics, see the Amazon Simple Notification Service Developer Guide.

  **Note**
  The Amazon SNS topic you choose must be in the same AWS region as the Amazon SES endpoint you use to receive email.
Writing your Lambda function

To process your email, your Lambda function can be invoked asynchronously (that is, using the Event invocation type). The event object passed to your Lambda function will contain metadata pertaining to the inbound email event. You can also use the metadata to access the message content from your Amazon S3 bucket.

If you want to actually control the mail flow, your Lambda function must be invoked synchronously (that is, using the RequestResponse invocation type) and your Lambda function must call the callback method with two arguments: the first argument is null, and the second argument is a disposition property that is set to either STOP_RULE, STOP_RULE_SET, or CONTINUE. If the second argument is null or does not have a valid disposition property, the mail flow continues and further actions and rules are processed, which is the same as with CONTINUE.

For example, you can stop the receipt rule set by writing the following line at the end of your Lambda function code:

```javascript
callback( null, { "disposition" : "STOP_RULE_SET" });
```

For AWS Lambda code samples, see Lambda function examples (p. 227). For examples of high-level use cases, see Use case examples (p. 224).

Input format

Amazon SES passes information to the Lambda function in JSON format. The top-level object contains a Records array, which is populated with properties eventSource, eventVersion, and ses. The ses object contains receipt and mail objects, which are in exactly the same format as in the Amazon SNS notifications described in Notification contents (p. 238).

**Note**

The data that Amazon SES passes to Lambda includes metadata about the message, as well as several email headers. However, it doesn't contain the body of the message.

The following is a high-level view of the structure of the input that Amazon SES provides to the Lambda function.

```json
{
  "Records": [
    {
      "eventSource": "aws:ses",
      "eventVersion": "1.0",
      "ses": {
        "receipt": {
          <same contents as SNS notification>
        },
        "mail": {
          <same contents as SNS notification>
        }
      }
    }
  ]
}
```

Return values

Your Lambda function can control mail flow by returning one of the following values:

- **STOP_RULE**—No further actions in the current receipt rule will be processed, but further receipt rules can be processed.
• **STOP_RULE_SET**—No further actions or receipt rules will be processed.
• **CONTINUE** or any other invalid value—This means that further actions and receipt rules can be processed.

Use case examples

The following examples outline some rules that you might set up to use Lambda function outcomes to control your mail flow. For demonstration purposes, many of these examples use the S3 action as the outcome.

**Use case 1: Drop spam across all domains**

This example demonstrates a global rule that drops spam across all of your domains. Rules 2 and 3 are included to show that you can apply domain-specific rules after the spam is dropped over all the domains.

**Rule 1**

*Recipient list*: Empty. This rule will therefore apply to all recipients under all of your verified domains.

*Actions*

1. Lambda action (synchronous) that returns **STOP_RULE_SET** if the email is spam. Otherwise, it returns **CONTINUE**. See the example Lambda function for dropping spam in Lambda function examples (p. 227).

**Rule 2**

*Recipient list*: example1.com

*Actions*

1. Any action.

**Rule 3**

*Recipient list*: example2.com

*Actions*

1. Any action.

**Use case 2: Bounce spam across all domains**

This example demonstrates a global rule that bounces spam across all of your domains. Rules 2 and 3 are included to show that you can apply domain-specific rules after the spam is bounced over all the domains.

**Rule 1**

*Recipient list*: Empty. This rule will therefore apply to all recipients under all of your verified domains.

*Actions*

1. Lambda action (synchronous) that returns **CONTINUE** if the email is spam. Otherwise, it returns **STOP_RULE**.
2. Bounce action ("500 5.6.1. Message content rejected").
3. Stop action.

**Rule 2**

*Recipient list: example1.com*

*Actions*

1. Any action

**Rule 3**

*Recipient list: example2.com*

*Actions*

1. Any action

**Use case 3: Apply the most specific rule**

This example demonstrates how you can use the Stop action to prevent emails from being processed by multiple rules. In this example, you have one rule for a specific address, and another rule for all email addresses under the domain. By using the Stop action, messages that match the rule for the specific email address are not processed by the more generic rule that applies to the domain.

**Rule 1**

*Recipient list: user@example.com*

*Actions*

1. Lambda action (asynchronous).
2. Stop action.

**Rule 2**

*Recipient list: example.com*

*Actions*

1. Any action.

**Use case 4: Log mail events to CloudWatch**

This example demonstrates how to keep an audit log of all mail going through your system before saving the mail to Amazon SES.

**Rule 1**

*Recipient list: example.com*

*Actions*

1. Lambda action (asynchronous) that writes the event object to a CloudWatch log. The example Lambda functions in *Lambda function examples (p. 227)* log to CloudWatch.
2. S3 action.

**Use case 5: Drops mail that fails DKIM**

This example demonstrates how you can save all incoming email to an Amazon S3 bucket, but only send email that goes to a specific email address, and passes DKIM, to your automated email application.

**Rule 1**

*Recipient list: example.com*

*Actions*

1. S3 action.
2. Lambda action (synchronous) that returns STOP_RULE_SET if the message fails DKIM. Otherwise, it returns CONTINUE.

**Rule 2**

*Recipient list: support@example.com*

*Actions*

1. Lambda action (asynchronous) that triggers the automated application.

**Use case 6: Filters mail based on subject line**

This example demonstrates how you can drop all of a domain's incoming mail that contains the word "discount" in the subject line, and then process mail intended for an automated system one way, and process mail addressed to all other recipients in the domain a different way.

**Rule 1**

*Recipient list: example.com*

*Actions*

1. Lambda action (synchronous) that returns STOP_RULE_SET if the subject line contains the word "discount". Otherwise, it returns CONTINUE.

**Rule 2**

*Recipient list: support@example.com*

*Actions*

1. S3 action with bucket 1.
2. Lambda action (asynchronous) that triggers the automated application.
3. Stop action.

**Rule 3**

*Recipient list: example.com*

*Actions*

1. S3 action with bucket 2.
2. Lambda action (asynchronous) that processes email for the rest of the domain.

Lambda function examples

This topic contains examples of Lambda functions that control mail flow.

Example 1: Drop spam

This example stops processing messages that have at least one spam indicator.

```javascript
exports.handler = function(event, context, callback) {
  console.log('Spam filter');

  var sesNotification = event.Records[0].ses;
  console.log("SES Notification:\n", JSON.stringify(sesNotification, null, 2));

  // Check if any spam check failed
  if (sesNotification.receipt.spfVerdict.status === 'FAIL'
      || sesNotification.receipt.dkimVerdict.status === 'FAIL'
      || sesNotification.receipt.spamVerdict.status === 'FAIL'
      || sesNotification.receipt.virusVerdict.status === 'FAIL') {
    console.log('Dropping spam');
    // Stop processing rule set, dropping message
    callback(null, {'disposition':'STOP_RULE_SET'});
  } else {
    callback(null, null);
  }
};
```

Example 2: Continue if a particular header is found

This example continues processing the current rule only if the email contains a specific header value.

```javascript
exports.handler = function(event, context, callback) {
  console.log('Header matcher');

  var sesNotification = event.Records[0].ses;
  console.log("SES Notification:\n", JSON.stringify(sesNotification, null, 2));

  // Iterate over the headers
  for (var index in sesNotification.mail.headers) {
    var header = sesNotification.mail.headers[index];
    // Examine the header values
    if (header.name === 'X-Header' && header.value === 'X-Value') {
      console.log('Found header with value.');
      callback(null, null);
      return;
    }
  }
  // Stop processing the rule if the header value wasn't found
  callback(null, {'disposition':'STOP_RULE_SET'});
};
```

Example 3: Retrieve email from Amazon S3

This example gets the raw email from Amazon S3 and processes it.

**Note**

You must first write the email to Amazon S3 using an S3 Action.
Example 4: Bounce messages that fail DMARC authentication

This examples sends a bounce message if an incoming email fails DMARC authentication.

**Note**
When using this example, set the value of the `emailDomain` environment variable to your email receiving domain.

```javascript
'use strict';
const AWS = require('aws-sdk');

// Assign the emailDomain environment variable to a constant.
const emailDomain = process.env.emailDomain;

exports.handler = (event, context, callback) => {
  console.log('Spam filter starting');

  const sesNotification = event.Records[0].ses;
  const messageId = sesNotification.mail.messageId;
  const receipt = sesNotification.receipt;
  console.log('Processing message:', messageId);

  if (receipt.dmarcVerdict.status === 'FAIL' && receipt.dmarcPolicy.status === 'REJECT') {
    // The values that make up the body of the bounce message.
    const sendBounceParams = {
      BounceSender: `mailer-daemon@${emailDomain}`
    };
    // Bounce the email.
    callback(null, null);
  }
};
```
ExtensionFields: [],
},
// Include custom text explaining why the email was bounced.
Explanation: "Unauthenticated email is not accepted due to the sending domain's
DMARC policy."

BouncedRecipientInfoList: receipt.recipients.map((recipient) => (
  Recipient: recipient,
  // Bounce with 550 5.6.1 Message content rejected
  BounceType: 'ContentRejected',
  )
}),

console.log('Bouncing message with parameters:');
console.log(JSON.stringify(sendBounceParams, null, 2));
// Try to send the bounce.
new AWS.SES().sendBounce(sendBounceParams, (err, data) => {
  // If something goes wrong, log the issue.
  if (err) {
    console.log(`An error occurred while sending bounce for message:
${messageId}`, err);
    callback(err);
  // Otherwise, log the message ID for the bounce email.
  } else {
    console.log(`Bounce for message ${messageId} sent, bounce message ID: ${data.MessageId}`);
    // Stop processing additional receipt rules in the rule set.
    callback(null, {
      disposition: 'stop_rule_set',
    });
  }
});
// If the DMARC verdict is anything else (PASS, QUARANTINE or GRAY), accept
// the message and process remaining receipt rules in the rule set.
} else {
  console.log('Accepting message:', messageId);
  callback();
}

S3 action

The S3 action delivers the mail to an Amazon S3 bucket and, optionally, notifies you through Amazon SNS. This action has the following options.

- **S3 Bucket**—The name of the Amazon S3 bucket to which to save received emails. You can also create a new Amazon S3 bucket when you set up your action by choosing Create S3 Bucket. Amazon SES provides you the raw, unmodified email, which is typically in Multipurpose Internet Mail Extensions (MIME) format. For more information about MIME format, see RFC 2045.

  **Important**
  When you save your emails to an Amazon S3 bucket, the default maximum email size (including headers) is 30 MB.

- **Object Key Prefix**—A key name prefix to use within the Amazon S3 bucket. Key name prefixes enable you to organize your Amazon S3 bucket in a folder structure. For example, if you use Email as your Object Key Prefix, your emails will appear in your Amazon S3 bucket in a folder named Email.

- **KMS Key (if "Encrypt Message" is selected in the Amazon SES console)**—The customer master key that Amazon SES should use to encrypt your emails before saving them to the Amazon S3 bucket. You can use the default master key or a custom master key you created in AWS KMS.

  **Note**
  The master key you choose must be in the same AWS region as the Amazon SES endpoint you use to receive email.
• To use the default master key, choose `aws/ses` when you set up the receipt rule in the Amazon SES console. If you use the Amazon SES API, you can specify the default master key by providing an ARN in the form of `arn:aws:kms:REGION:AWSACCOUNTID:alias/aws/ses`. For example, if your AWS account ID is 123456789012 and you want to use the default master key in the US West (Oregon) region, the ARN of the default master key would be `arn:aws:kms:us-west-2:123456789012:alias/aws/ses`. If you use the default master key, you don’t need to perform any extra steps to give Amazon SES permission to use the key.

• To use a custom master key you created in AWS KMS, provide the ARN of the master key and ensure that you add a statement to your key's policy to give Amazon SES permission to use it. For more information about giving permissions, see Giving permissions to Amazon SES for email receiving (p. 215).

For more information about using AWS KMS with Amazon SES, see the AWS Key Management Service Developer Guide. If you do not specify a master key in the console or API, Amazon SES will not encrypt your emails.

Important
Your mail is encrypted by Amazon SES using the Amazon S3 encryption client before the mail is submitted to Amazon S3 for storage. It is not encrypted using Amazon S3 server-side encryption. This means that you must use the Amazon S3 encryption client to decrypt the email after retrieving it from Amazon S3, as the service has no access to use your AWS KMS keys for decryption. This encryption client is available in the AWS SDK for Java and the AWS SDK for Ruby. For more information about client-side encryption using AWS KMS master keys, see the Amazon Simple Storage Service Developer Guide.

• **SNS Topic**—The name or ARN of the Amazon SNS topic to notify when an email is saved to the Amazon S3 bucket. An example of an Amazon SNS topic ARN is `arn:aws:sns:us-west-2:123456789012:MyTopic`. You can also create an Amazon SNS topic when you set up your action by choosing Create SNS Topic. For more information about Amazon SNS topics, see the Amazon Simple Notification Service Developer Guide.

Note
The Amazon SNS topic you choose must be in the same AWS region as the Amazon SES endpoint you use to receive email.

### SNS action

The SNS action publishes the mail using an Amazon SNS notification. The notification includes the complete email content. This action has the following options.

• **SNS Topic**—The name or ARN of the Amazon SNS topic to which to publish the emails. The Amazon SNS notifications will contain a raw, unmodified copy of the email, which is typically in Multipurpose Internet Mail Extensions (MIME) format. For more information about MIME format, see RFC 2045.

Important
If you choose to receive your emails through Amazon SNS notifications, the maximum email size (including headers) is 150 KB. Larger emails will bounce. If you anticipate emails larger than this size, save the emails to an Amazon S3 bucket instead.

An example of an Amazon SNS topic ARN is `arn:aws:sns:us-west-2:123456789012:MyTopic`. You can also create an Amazon SNS topic when you set up your action by choosing Create SNS Topic. For more information about Amazon SNS topics, see the Amazon Simple Notification Service Developer Guide.

Note
The Amazon SNS topic you choose must be in the same AWS region as the Amazon SES endpoint you use to receive email.

• **Encoding**—The encoding to use for the email within the Amazon SNS notification. UTF-8 is easier to use, but may not preserve all special characters when a message was encoded with a different
encoding format. Base64 preserves all special characters. For information about UTF-8 and Base64, see RFC 3629 and RFC 4648, respectively.

Stop action

The Stop action terminates the evaluation of the receipt rule set and, optionally, notifies you through Amazon SNS. This action has the following options.

- **SNS Topic**—The name or ARN of the Amazon SNS topic to notify when the Stop action is performed. An example of an Amazon SNS topic ARN is `arn:aws:sns:us-west-2:123456789012:MyTopic`. You can also create an Amazon SNS topic when you set up your action by choosing **Create SNS Topic**. For more information about Amazon SNS topics, see the Amazon Simple Notification Service Developer Guide.

  **Note**
  The Amazon SNS topic you choose must be in the same AWS region as the Amazon SES endpoint you use to receive email.

WorkMail action

The WorkMail action integrates with Amazon WorkMail. If Amazon WorkMail performs all of your email processing, you will typically not use this action directly because Amazon WorkMail takes care of the setup. This action has the following options.

- **Organization ARN**—The ARN of the Amazon WorkMail organization. Amazon WorkMail organization ARNs are in the form `arn:aws:workmail:region:account_ID:organization/organization_ID`, where:
  - **region** is the region in which you are using Amazon SES and Amazon WorkMail. (You must use them from the same region.) An example is us-west-2.
  - **account_ID** is the AWS account ID. You can find your AWS account ID on the Account page of the AWS Management Console.
  - **organization_ID** is a unique identifier that Amazon WorkMail generates when you create an organization. You can find the organization ID in the Amazon WorkMail console on the Organization Settings page of your organization.

  An example of a complete Amazon WorkMail organization ARN is `arn:aws:workmail:us-west-2:123456789012:organization/m-68755160c4cb4e29a2b2f8fb58f359d7`. For information about Amazon WorkMail organizations, see the Amazon WorkMail Administrator Guide.

- **SNS Topic**—The name or ARN of the Amazon SNS topic to notify when the Amazon WorkMail action is taken. An example of an Amazon SNS topic ARN is `arn:aws:sns:us-west-2:123456789012:MyTopic`. You can also create an Amazon SNS topic when you set up your action by choosing **Create SNS Topic**. For more information about Amazon SNS topics, see the Amazon Simple Notification Service Developer Guide.

  **Note**
  The Amazon SNS topic you choose must be in the same AWS region as the Amazon SES endpoint you use to receive email.

Managing email receiving in Amazon SES

After you create your receipt rule sets, receipt rules, and IP address filters, you can use the Amazon SES console or API to edit, delete, and perform other operations. You can also examine the Amazon SES notifications you receive, and use Amazon CloudWatch to view your error metrics.

Topics in this section:
Managing receipt rule sets for Amazon SES email receiving

After you create a receipt rule set as described in Creating a receipt rule set (p. 218), you can update it as needed. Although editing a receipt rule set usually consists of editing individual receipt rules as described in Managing receipt rules (p. 234), you can also delete, activate, disable, and copy receipt rule sets. Additionally, you can reorder the receipt rules in a receipt rule set. These operations are described in the following sections.

Topics in this section:
- Deleting a receipt rule set (p. 232)
- Activating and disabling a receipt rule set (p. 232)
- Copying a receipt rule set (p. 233)
- Reordering receipt rules (p. 234)

Deleting a receipt rule set

You can use the Amazon SES console or the DeleteReceiptRuleSet API to delete a receipt rule set.

Note
You cannot delete the receipt rule set that is currently active.

To delete a receipt rule set (console)

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under Email Receiving, choose Rule Sets.
3. In the Inactive Rule Sets list, select the receipt rule set that you want to delete.
4. From the Actions menu, choose Delete, and then confirm that you want to delete the receipt rule set.

For information about how to use the DeleteReceiptRuleSet API to delete a receipt rule set, see the Amazon Simple Email Service API Reference.

Activating and disabling a receipt rule set

Each receipt rule set is in one of two states: active or disabled. Only one of your receipt rule sets can be active at any given time. Disabled receipt rule sets can be useful in cases where you want to make changes to your active receipt rule set, but you do not want those changes to be active until you are sure your updates are correct. In that case, you can copy the active receipt rule set and make changes to the copied, disabled receipt rule set. After you're satisfied with the changes, you can activate the copied receipt rule set. When you activate a receipt rule set, all other receipt rule sets are disabled automatically.

Note
To disable email receiving through Amazon SES completely, disable all of your receipt rule sets.
You can use the Amazon SES console or the `SetActiveReceiptRuleSet` API to control which rule set is active.

**To activate a disabled receipt rule set (console)**

1. Sign in to the AWS Management Console and open the Amazon SES console at `https://console.aws.amazon.com/ses/`.
2. In the left navigation pane, under **Email Receiving**, choose **Rule Sets**.
3. In the **Inactive Rule Sets** list, select the receipt rule set that you want to activate.
4. Choose **Set as Active Rule Set**.

**To disable the active receipt rule set (console)**

1. Sign in to the AWS Management Console and open the Amazon SES console at `https://console.aws.amazon.com/ses/`.
2. In the left navigation pane, under **Email Receiving**, choose **Rule Sets**.
3. Under **Active Rule Set**, choose **Disable Active Rule Set**, and then confirm that you want to disable the receipt rule set.

For information about how to use the `SetActiveReceiptRuleSet` API to activate or disable a rule set, see the Amazon Simple Email Service API Reference.

**Copying a receipt rule set**

You can use the Amazon SES console or the `CloneReceiptRuleSet` API to copy a receipt rule set. If you use the Amazon SES console, the procedure differs slightly, depending on whether the receipt rule set you want to copy is active or disabled.

**To copy the active receipt rule set (console)**

1. Sign in to the AWS Management Console and open the Amazon SES console at `https://console.aws.amazon.com/ses/`.
2. In the left navigation pane, under **Email Receiving**, choose **Rule Sets**.
3. In the content pane, choose **Copy Active Rule Set**.
4. In the **Copy Rule Set** dialog box, type the name you want to assign to the copied receipt rule set.
5. Choose **Copy Rule Set**. The copied receipt rule set will appear in the **Inactive Rule Sets** list.

**To copy a disabled receipt rule set (console)**

1. Sign in to the AWS Management Console and open the Amazon SES console at `https://console.aws.amazon.com/ses/`.
2. In the left navigation pane, under **Email Receiving**, choose **Rule Sets**.
3. In the **Inactive Rule Sets** list, select the receipt rule set that you want to copy.
4. From the **Actions** menu, choose **Copy**.
5. In the **Copy Rule Set** dialog box, type the name you want to assign to the copied receipt rule set.
6. Choose **Copy Rule Set**. The copied receipt rule set will appear in the **Inactive Rule Sets** list.

For information about how to use the `CloneReceiptRuleSet` API to copy a receipt rule set, see the Amazon Simple Email Service API Reference.
Reordering receipt rules

You can use the Amazon SES console or the ReorderReceiptRuleSet API to reorder receipt rules in a receipt rule set. If you use the Amazon SES console, the procedure differs slightly, depending on whether the receipt rule set is active or disabled.

To reorder receipt rules in the active receipt rule set (console)
1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under Email Receiving, choose Rule Sets.
3. In the content pane, choose View Active Rule Set.
5. Use the up and down arrows next to the receipt rule names to reorder the receipt rules, and then choose Save Order.

To reorder receipt rules in a disabled receipt rule set (console)
1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under Email Receiving, choose Rule Sets.
3. In the Inactive Rule Sets list, select the receipt rule set.
5. Use the up and down arrows next to the receipt rule names to reorder the receipt rules, and then choose Save Order.

For information about how to use the ReorderReceiptRuleSet API to reorder receipt rules in a receipt rule set, see the Amazon Simple Email Service API Reference.

Managing receipt rules for Amazon SES email receiving

In addition to creating receipt rules as described in Creating receipt rules (p. 219), you can edit, delete, enable, disable, copy, and set the position of a receipt rule in its receipt rule set, as described in the following sections.

Note
The instructions in this section assume that the receipt rule is in the active receipt rule set. To edit the receipt rules of a disabled receipt rule set, choose a receipt rule set from the Inactive Rule Sets list. From there, the instructions for editing receipt rules are the same as for the active receipt rule set.

Topics in this section:
- Editing a receipt rule (p. 235)
- Deleting a receipt rule (p. 235)
- Enabling and disabling a receipt rule (p. 235)
- Copying a receipt rule (p. 236)
- Setting the position of a receipt rule (p. 236)
Managing receipt rules

Editing a receipt rule

You can use the Amazon SES console or the Amazon SES API to edit a receipt rule. It is easier to use the Amazon SES console.

**To edit a receipt rule (console)**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under Email Receiving, choose Rule Sets.
3. In the content pane, choose View Active Rule Set or choose a receipt rule set from the Inactive Rule Sets list.
4. In the details pane, choose the receipt rule you want to edit.
5. In the Edit Rule pane, edit the policy, and then choose Save Rule.

If you want to use the Amazon SES API instead, use the DescribeReceiptRule API to retrieve the rule, use a text editor to edit the rule, and then use the UpdateReceiptRule API to overwrite the previous version of the rule. For more information, see the Amazon Simple Email Service API Reference.

Deleting a receipt rule

You can use the Amazon SES console or the DeleteReceiptRule API to delete a receipt rule.

**To delete a receipt rule (console)**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under Email Receiving, choose Rule Sets.
3. In the content pane, choose View Active Rule Set or choose a receipt rule set from the Inactive Rule Sets list.
4. In the details pane, select the receipt rule.
5. From the Actions menu, choose Delete, and then confirm that you want to delete the receipt rule.

For information about how to use the DeleteReceiptRule API to delete a rule, see the Amazon Simple Email Service API Reference.

Enabling and disabling a receipt rule

You can use the Amazon SES console or the Amazon SES API to enable or disable a receipt rule. It is easier to use the Amazon SES console.

**To enable or disable a receipt rule (console)**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under Email Receiving, choose Rule Sets.
3. In the content pane, choose View Active Rule Set or choose a receipt rule set from the Inactive Rule Sets list.
4. In the details pane, choose the receipt rule you want to edit.
5. In the Edit Rule pane, select or clear Enabled, and then choose Save Rule.
If you want to use the Amazon SES API instead, you can use the DescribeReceiptRule API to retrieve the receipt rule, use a text editor to edit the receipt rule's Enabled field, and then use the UpdateReceiptRule API to overwrite the previous version of the receipt rule. For more information, see the Amazon Simple Email Service API Reference.

**Copying a receipt rule**

You can use the Amazon SES console or the Amazon SES API to copy a receipt rule. It is easier to use the Amazon SES console.

**To copy a receipt rule (console)**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under Email Receiving, choose Rule Sets.
3. In the content pane, choose View Active Rule Set or choose a receipt rule set from the Inactive Rule Sets list.
4. In the details pane, select the receipt rule.
5. From the Actions menu, choose Copy Rule.
6. In the Copy Rule dialog box, type a new receipt rule name and select the destination receipt rule set. The new receipt rule will be inserted at the beginning of the receipt rule set, and it will initially be disabled.

If you want to use the Amazon SES API instead, you can use the DescribeReceiptRule API to retrieve the receipt rule, use a text editor to edit the receipt rule's name and receipt rule set (if desired), and then pass that receipt rule to the CreateReceiptRule API. For more information, see the Amazon Simple Email Service API Reference.

**Setting the position of a receipt rule**

You can use the Amazon SES console or the SetReceiptRulePosition API to change the position of a receipt rule in the receipt rule set.

**To set the position of a receipt rule (console)**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under Email Receiving, choose Rule Sets.
3. In the content pane, choose View Active Rule Set or choose a receipt rule set from the Inactive Rule Sets list.
4. In the content pane, choose Reorder Rules.
5. Use the up and down arrows next to the receipt rule names to reorder the receipt rules, and then choose Save Order.

For information about how to use the SetReceiptRulePosition API to change the position of a receipt rule in the receipt rule set, see the Amazon Simple Email Service API Reference.

**Managing IP address filters for Amazon SES email receiving**

In addition to creating IP address filters as explained in Creating IP address filters (p. 218), you can view and delete them, as described in the following sections.
Viewing IP address filters

You can use the Amazon SES console or the ListReceiptFilters API to get a list of your IP address filters.

To view your IP address filters (console)
1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under Email Receiving, choose IP Address Filters. You will see a list of your IP address filters.

For information about how to use the ListReceiptFilters API to get a list of your IP address filters, see the Amazon Simple Email Service API Reference.

Deleting an IP address filter

You can use the Amazon SES console or the DeleteReceiptFilter API to delete an IP address filter.

To delete an IP address filter (console)
1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, under Email Receiving, choose IP Address Filters.
3. In the details pane, select the IP address filter.
4. Choose Delete, and then confirm that you want to delete the IP address filter.

For information about how to use the DeleteReceiptFilter API to delete an IP address filter, see the Amazon Simple Email Service API Reference.

Viewing metrics for Amazon SES email receiving

You can use Amazon CloudWatch (CloudWatch) to view failure metrics for your receipt rules. You'll find the metrics under SES/Rule Metrics.

There are two failure metrics:

- **PublishFailure** – Amazon SES encountered an error when it tried to execute the actions you configured.
- **PublishExpired** – Amazon SES encountered an error when it tried to execute the actions you configured, and Amazon SES will no longer retry to deliver the email. This failure can be permanent or transient. Amazon SES will no longer retry because the action did not succeed within four hours.

These errors can occur, for example, if you deleted or revoked permissions to an Amazon S3 bucket, Amazon SNS topic, or Lambda function that an action in one of your receipt rules was configured to use.

Important
Changes you make to fix your receipt rule set will apply only to emails that Amazon SES receives after the update. Emails are always evaluated against the receipt rule set that was in place at the time the email was received.

The following figure shows the metrics in the CloudWatch console.
Using notifications for Amazon SES email receiving

When you receive an email, Amazon SES executes the rules in the active receipt rule set. You can configure receipt rules to send you notifications using Amazon SNS. Your receipt rules can send two different types of notifications:

- **Notifications sent from SNS actions** – When you add an SNS (p. 230) action to a receipt rule, it sends information about the email. If the message is 150KB or smaller, this notification type also includes the complete MIME body of the email.

- **Notifications sent from other action types** – When you add any other action type (including Bounce (p. 221), Lambda (p. 222), Stop Rule Set (p. 231), or WorkMail (p. 231) actions) to a receipt rule, you can optionally specify an Amazon SNS topic. If you do, you will receive notifications when these actions are performed. These notifications contain information about the email, but do not contain the content of the email.

This section describes the contents of these notifications, and provides an example of each type of notification.

**Topics in this section:**
- Contents of notifications for Amazon SES email receiving (p. 238)
- Examples of notifications for Amazon SES email receiving (p. 244)

**Contents of notifications for Amazon SES email receiving**

All notifications for email receiving are published to Amazon Simple Notification Service (Amazon SNS) topics in JavaScript Object Notation (JSON) format.

**Top-level JSON object**

The top-level JSON object contains the following fields.
### Using notifications

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notificationType</td>
<td>The notification type. For this type of notification, the value is always Received.</td>
</tr>
<tr>
<td>receipt (p. 239)</td>
<td>Object that contains information about the email delivery.</td>
</tr>
<tr>
<td>mail (p. 243)</td>
<td>Object that contains information about the email associated with the notification.</td>
</tr>
</tbody>
</table>
| content            | String that contains the raw, unmodified email, which is typically in Multipurpose Internet Mail Extensions (MIME) format. For more information about MIME format, see RFC 2045.  
  **Note**  
  This field is present only if the notification was triggered by an SNS action. Notifications triggered by all other actions do not contain this field. |

**receipt object**

The `receipt` object has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action (p. 240)</td>
<td>Object that encapsulates information about the action that was executed. For a list of possible values, see <code>action object (p. 240)</code>.</td>
</tr>
<tr>
<td>dkimVerdict (p. 241)</td>
<td>Object that indicates whether the DomainKeys Identified Mail (DKIM) check passed. For a list of possible values, see <code>dkimVerdict object (p. 241)</code>.</td>
</tr>
</tbody>
</table>
| dmarcPolicy        | Indicates the Domain-based Message Authentication, Reporting & Conformance (DMARC) settings for the sending domain. This field only appears if the message fails DMARC authentication.  
  Possible values for this field are:  
  • **none**: The owner of the sending domain requests that no specific action be taken on messages that fail DMARC authentication.  
  • **quarantine**: The owner of the sending domain requests that messages that fail DMARC authentication be treated by receivers as suspicious.  
  • **reject**: The owner of the sending domain requests that messages that fail DMARC authentication be rejected. |
<p>| dmarcVerdict (p. 241) | Object that indicates whether the Domain-based Message Authentication, Reporting |</p>
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp; Conformance (DMARC) check passed. For a list of possible values, see dmarcVerdict object (p. 241).</td>
<td></td>
</tr>
<tr>
<td>processingTimeMillis</td>
<td>String that specifies the period, in milliseconds, from the time Amazon SES received the message to the time it triggered the action.</td>
</tr>
<tr>
<td>recipients</td>
<td>A list of recipients (specifically, the envelope RCPT TO addresses) that were matched by the active receipt rule (p. 219). The addresses listed here may differ from those listed by the destination field in the the section called &quot;mail object&quot; (p. 243).</td>
</tr>
<tr>
<td>spamVerdict (p. 242)</td>
<td>Object that indicates whether the message is spam. For a list of possible values, see spamVerdict object (p. 242).</td>
</tr>
<tr>
<td>spfVerdict (p. 242)</td>
<td>Object that indicates whether the Sender Policy Framework (SPF) check passed. For a list of possible values, see spfVerdict object (p. 242).</td>
</tr>
<tr>
<td>timestamp</td>
<td>String that specifies the qualified date and time at which the action was triggered, in ISO 8601 format.</td>
</tr>
<tr>
<td>virusVerdict (p. 243)</td>
<td>Object that indicates whether the message contains a virus. For a list of possible values, see virusVerdict object (p. 243).</td>
</tr>
</tbody>
</table>

**action object**

The action object has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>String that indicates the type of action that was executed. Possible values are S3, SNS, Bounce, Lambda, Stop, and WorkMail.</td>
</tr>
<tr>
<td>topicArn</td>
<td>String that contains the Amazon Resource Name (ARN) of the Amazon SNS topic to which the notification was published.</td>
</tr>
<tr>
<td>bucketName</td>
<td>String that contains the name of the Amazon S3 bucket to which the message was published. Present only for the S3 action type.</td>
</tr>
<tr>
<td>objectKey</td>
<td>String that contains a name that uniquely identifies the email in the Amazon S3 bucket. This is the same as the messageId in the the section called &quot;mail object&quot; (p. 243). Present only for the S3 action type.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>smtpReplyCode</td>
<td>String that contains the SMTP reply code, as defined by RFC 5321. Present only for the bounce action type.</td>
</tr>
<tr>
<td>statusCode</td>
<td>String that contains the SMTP enhanced status code, as defined by RFC 3463. Present only for the bounce action type.</td>
</tr>
<tr>
<td>message</td>
<td>String that contains the human-readable text to include in the bounce message. Present only for the bounce action type.</td>
</tr>
<tr>
<td>sender</td>
<td>String that contains the email address of the sender of the email that bounced. This is the address from which the bounce message was sent. Present only for the bounce action type.</td>
</tr>
<tr>
<td>functionArn</td>
<td>String that contains the ARN of the Lambda function that was triggered. Present only for the Lambda action type.</td>
</tr>
<tr>
<td>invocationType</td>
<td>String that contains the invocation type of the Lambda function. Possible values are RequestResponse and Event. Present only for the Lambda action type.</td>
</tr>
<tr>
<td>organizationArn</td>
<td>String that contains the ARN of the Amazon WorkMail organization. Present only for the WorkMail action type.</td>
</tr>
</tbody>
</table>

**dkimVerdict object**

The dkimVerdict object has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| status     | String that contains the DKIM verdict. Possible values are:  
- PASS: The message passed DKIM authentication.  
- FAIL: The message failed DKIM authentication.  
- GRAY: The message is not DKIM-signed.  
- PROCESSING_FAILED: There is an issue that prevents Amazon SES from checking the DKIM signature. For example, DNS queries are failing or the DKIM signature header is not formatted properly. |

**dmarcVerdict object**

The dmarcVerdict object has the following fields.
### Using notifications

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>String that contains the DMARC verdict. Possible values are:</td>
</tr>
<tr>
<td></td>
<td>- PASS: The message passed DMARC authentication.</td>
</tr>
<tr>
<td></td>
<td>- FAIL: The message failed DMARC authentication.</td>
</tr>
<tr>
<td></td>
<td>- GRAY: The message failed DMARC authentication, and the sending domain does not have a DMARC policy, or uses the p=none policy.</td>
</tr>
<tr>
<td></td>
<td>- PROCESSING_FAILED: There is an issue that prevents Amazon SES from providing a DMARC verdict.</td>
</tr>
</tbody>
</table>

#### spamVerdict object

The `spamVerdict` object has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>String that contains the result of spam scanning. Possible values are:</td>
</tr>
<tr>
<td></td>
<td>- PASS: The spam scan determined that the message is unlikely to contain spam.</td>
</tr>
<tr>
<td></td>
<td>- FAIL: The spam scan determined that the message is likely to contain spam.</td>
</tr>
<tr>
<td></td>
<td>- GRAY: Amazon SES scanned the email but could not determine with confidence whether it is spam.</td>
</tr>
<tr>
<td></td>
<td>- PROCESSING_FAILED: Amazon SES was unable to scan the email. For example, the email is not a valid MIME message.</td>
</tr>
</tbody>
</table>

#### spfVerdict object

The `spfVerdict` object has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>String that contains the SPF verdict. Possible values are:</td>
</tr>
<tr>
<td></td>
<td>- PASS: The message passed SPF authentication.</td>
</tr>
<tr>
<td></td>
<td>- FAIL: The message failed SPF authentication.</td>
</tr>
<tr>
<td></td>
<td>- GRAY: There is no SPF policy under the domain used in the MAIL FROM command.</td>
</tr>
</tbody>
</table>
Using notifications

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROCESSING_FAILED</td>
<td>There is an issue that prevents Amazon SES from checking the SPF record. For example, DNS queries are failing.</td>
</tr>
</tbody>
</table>

virusVerdict object

The virusVerdict object has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>String that contains the result of virus scanning. Possible values are:</td>
</tr>
<tr>
<td>PASS</td>
<td>The message does not contain a virus.</td>
</tr>
<tr>
<td>FAIL</td>
<td>The message contains a virus.</td>
</tr>
<tr>
<td>GRAY</td>
<td>Amazon SES scanned the email but could not determine with confidence whether it contains a virus.</td>
</tr>
<tr>
<td>PROCESSING_FAILED</td>
<td>Amazon SES is unable to scan the content of the email. For example, the email is not a valid MIME message.</td>
</tr>
</tbody>
</table>

mail object

The mail object has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>destination</td>
<td>A complete list of all recipient addresses (including To: and CC: recipients) from the MIME headers of the incoming email.</td>
</tr>
<tr>
<td>messageId</td>
<td>String that contains the unique ID assigned to the email by Amazon SES. If the email was delivered to Amazon S3, the message ID is also the Amazon S3 object key that was used to write the message to your Amazon S3 bucket.</td>
</tr>
<tr>
<td>source</td>
<td>String that contains the email address (specifically, the envelope MAIL FROM address) that the email was sent from.</td>
</tr>
<tr>
<td>timestamp</td>
<td>String that contains the time at which the email was received, in ISO8601 format.</td>
</tr>
<tr>
<td>headers</td>
<td>A list of Amazon SES headers and your custom headers. Each header in the list has a name field and a value field.</td>
</tr>
<tr>
<td>commonHeaders (p. 244)</td>
<td>A list of headers common to all emails. Each header in the list is composed of a name and a value.</td>
</tr>
</tbody>
</table>
Using notifications

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>headersTruncated</td>
<td>Specifies whether the headers were truncated in the notification, which happens if the headers are larger than 10 KB. Possible values are true and false.</td>
</tr>
</tbody>
</table>

**commonHeaders object**

The commonHeaders object can have the fields shown in the following table. The fields present in this object vary depending on which fields were present in the incoming email.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>messageId</td>
<td>The ID of the original message.</td>
</tr>
<tr>
<td>date</td>
<td>The date and time when Amazon SES received the message.</td>
</tr>
<tr>
<td>to</td>
<td>The values in the To header of the email.</td>
</tr>
<tr>
<td>cc</td>
<td>The values in the CC header of the email.</td>
</tr>
<tr>
<td>bcc</td>
<td>The values in the BCC header of the email.</td>
</tr>
<tr>
<td>from</td>
<td>The values in the From header of the email.</td>
</tr>
<tr>
<td>sender</td>
<td>The values in the Sender header of the email.</td>
</tr>
<tr>
<td>returnPath</td>
<td>The values in the Return-Path header of the email.</td>
</tr>
<tr>
<td>reply-to</td>
<td>The values in the Reply-To header of the email.</td>
</tr>
<tr>
<td>subject</td>
<td>The value of the Subject header for the email.</td>
</tr>
</tbody>
</table>

**Examples of notifications for Amazon SES email receiving**

This section includes examples of the following types of notifications:

- A notification sent as a result of an SNS action. (p. 244)
- A notification sent as a result of another type of action (p. 246) (an alert notification).

**Notification of an SNS action**

This section contains an example of an SNS action notification. Unlike the alert notification shown previously, it includes a content section that contains the email, which is typically in Multipurpose Internet Mail Extensions (MIME) format.

```json
{
  "notificationType":"Received",
  "receipt":{
    "timestamp":"2015-09-11T20:32:33.936Z",
    "processingTimeMillis":222,
    "recipients":[
      "recipient@example.com"
    ],
  }
}
```
"spamVerdict":{
  "status":"PASS"
},
"virusVerdict":{
  "status":"PASS"
},
"spfVerdict":{
  "status":"PASS"
},
"dkimVerdict":{
  "status":"PASS"
},
"action":{
  "type":"SNS",
  "topicArn":"arn:aws:sns:us-east-1:012345678912:example-topic"
}
},
"mail":{
  "timestamp":"2015-09-11T20:32:33.936Z",
  "source":"61967230-7A45-4A9D-BEC9-87C8CF2211C9@example.com",
  "messageId":"d6iitobk75ur44p8kdnnp7g2n800",
  "destination":[
    "recipient@example.com"
  ],
  "headersTruncated":false,
  "headers":{
    "name":"Return-Path",
    "value":"<0000014fbe1c09cf-7cb9f704-7531-4e53-89a1-5fa9744f5eb6-000000@amazonses.com>"
  },
  "name":"Received",
  "value":"from a9-183.smtp-out.amazonses.com (a9-183.smtp-out.amazonses.com [54.240.9.183]) by inbound-smtp.us-east-1.amazonaws.com with SMTP id d6iitobk75ur44p8kdnnp7g2n800 for recipient@example.com; Fri, 11 Sep 2015 20:32:33 +0000 (UTC)"
  },
  "name":"DKIM-Signature",
  "value":"v=1; a=rsa-sha256; q=dns/txt; c=relaxed/simple; s=ug7nbif4gcclmpwj322ax3p6ow6ysn; d=amazonses.com; t=1442003552; h=From:To:Subject:MIME-Version:Content-Type:Content-Transfer-Encoding:Date:Message-ID:Feedback-ID; bh=DWr3IOmYWoXCA9ARqGC/UaODghfffWfNRKri2Mckyt4=; b=4ukUDSFQghsibz+e2PODwlwp7oJzakrzupr6LBe6sUuqspBkgj56zUwc29rFbdJFhLx30v7DeYvN0N38tqswFs8tvaajXpQssXRC1c9W9x8x875J041rCLAJV7EGBLa25vPfx4hHst1XPyx9wmgdHLhmUuh8oEpVq66bHzzs7f;g="
  },
  "name":"From",
  "value":"sender@example.com"
  },
  "name":"To",
  "value":"recipient@example.com"
  },
  "name":"Subject",
  "value":"Example subject"
  },
  "name":"MIME-Version",
  "value":"1.0"
  },
  "name":"Content-Type",}
Alert notification

This section contains an example of an Amazon SNS notification that can be triggered by an S3 action. Notifications triggered by Lambda actions, bounce actions, stop actions, and WorkMail actions are similar. Although the notification contains information about the email, it does not contain the content of the email itself.

```json
{
    "notificationType": "Received",
    "value": "text/plain; charset=UTF-8",
    "name": "Content-Transfer-Encoding",
    "value": "7bit",
    "name": "Date",
    "value": "Fri, 11 Sep 2015 20:32:32 +0000",
    "name": "Message-ID",
    "value": "<61967230-7A45-4A9D-87C8C9211C9@example.com>"
}
```
"receipt": {
  "timestamp": "2015-09-11T20:32:33.936Z",
  "processingTimeMillis": 406,
  "recipient": [
    "recipient@example.com"
  ],
  "spamVerdict": {
    "status": "PASS"
  },
  "virusVerdict": {
    "status": "PASS"
  },
  "spfVerdict": {
    "status": "PASS"
  },
  "dkimVerdict": {
    "status": "PASS"
  },
  "action": {
    "type": "S3",
    "topicArn": "arn:aws:sns:us-east-1:012345678912:example-topic",
    "bucketName": "my-S3-bucket",
    "objectKey": "\email"
  }
},
"mail": {
  "timestamp": "2015-09-11T20:32:33.936Z",
  "source": "0000014fbe1c09cf-7cb9f704-7531-4e53-89a1-5fa9744f5eb6-000000@amazonses.com",
  "messageId": "d6iitobk75ur44p8kdnnp7g2n800",
  "destination": [
    "recipient@example.com"
  ],
  "headersTruncated": false,
  "headers": [
    {
      "name": "Return-Path",
      "value": "<0000014fbe1c09cf-7cb9f704-7531-4e53-89a1-5fa9744f5eb6-000000@amazonses.com>"
    },
    {
      "name": "Received",
      "value": "from a9-183.smtp-out.amazonses.com (a9-183.smtp-out.amazonses.com
[54.240.9.183]) by inbound-smtp.us-east-1.amazonaws.com with SMTP id
d6iitobk75ur44p8kdnnp7g2n800 for recipient@example.com; Fri, 11 Sep 2015 20:32:33 +0000
(UTC)"
    },
    {
      "name": "DKIM-Signature",
      "value": "v=1; a=rsa-sha256; q=dns/txt; c=relaxed/simple;
    s=amazonses.com; t=1442003552;
    h=From:To:Subject:MIME-Version:Content-Type:Content-Transfer-Encoding:Date:Message-ID:Feedback-ID:
    b=p4ukUDDSfQhqiub+bZP0D1Wlkp7oJZakrzuapr6LBe6SuuvqBKig56Uw292FbJFhI3oV7DeYV0N38s7qwsF8ivcajXpQsXRc1cW9z8x875J041rC1AjV7EGbLmdyVPvPX
4hnSt1Xpyx5wmgdHIhmUuh8o2KpVQfGisbHgzzf7g="
    },
    {
      "name": "From",
      "value": "sender@example.com"
    },
    {
      "name": "To",
      "value": "recipient@example.com"
    },
    {
      "name": "Subject",
      "value": "Example subject"
  ]
}
Amazon Simple Email Service Classic Developer Guide
Using notifications

`,

  "name": "MIME-Version",
  "value": "1.0"
},

  "name": "Content-Type",
  "value": "text/plain; charset=UTF-8"
},

  "name": "Content-Transfer-Encoding",
  "value": "7bit"
},

  "name": "Date",
  "value": "Fri, 11 Sep 2015 20:32:32 +0000"
},

  "name": "Message-ID",
  "value": "<61967230-7A45-4A9D-BEC9-87CBEF2211C9@example.com>"
},

  "name": "X-SES-Outgoing",
  "value": "2015.09.11-54.240.9.183"
},

  "name": "Feedback-ID",
  "value": "1.us-east-1.Krv2FKpFdWV+KUYw3Qd6wcpPJ4Sv/pOOpEjP5Hn2u2o4=:AmazonSES"
},

"commonHeaders": {
  "returnPath": "0000014fbe1c09cf-7cb9f704-7531-4e53-89a1-5fa9744f5eb6-000000@amazonses.com",
  "from": [
    "sender@example.com"
  ],
  "date": "Fri, 11 Sep 2015 20:32:32 +0000",
  "to": [
    "recipient@example.com"
  ],
  "messageId": "<61967230-7A45-4A9D-BEC9-87CBEF2211C9@example.com>",
  "subject": "Example subject"
}
Using Amazon SES configuration sets

Configuration sets are groups of rules that you can apply to the emails you send using Amazon SES. You apply a configuration set to an email by including a reference to the configuration set in the headers of the email. When you apply a configuration set to an email, all of the rules in that configuration set are applied to the email. For more information about specifying configuration sets in your emails, see Specifying a configuration set when you send email (p. 257).

You can use configuration sets to apply the following types of rules to your emails:

- **Event publishing** – Amazon SES can track the number of send, delivery, open, click, bounce, and complaint events for each email you send. You can use event publishing to send information about these events to other AWS services. For example, you can send your email metrics to an Amazon Kinesis Data Firehose destination, and then analyze it using Amazon Kinesis Data Analytics. Alternatively, you can send bounce and complaint information to Amazon SNS and receive notifications immediately when those events occur.

- **IP pool management** – If you lease dedicated IP addresses to use with Amazon SES, you can create groups of these addresses, called dedicated IP pools. You can then associate these dedicated IP pools with configuration sets. A common use case is to create one pool of dedicated IP addresses for sending marketing communications, and another for sending transactional emails. Your sender reputation for transactional emails is then isolated from that of your marketing emails.

Configuration sets can contain one, both, or neither of these types of rules.

To learn more about managing configuration sets and their related components, see the following topics:

- Managing configuration sets in Amazon SES (p. 249)
- Managing event destinations in Amazon SES (p. 254)
- Managing dedicated IP pools (p. 256)

Managing configuration sets in Amazon SES

This section contains procedures for creating configuration sets, viewing a list of your existing configuration sets, viewing the details of individual configuration sets, and deleting configuration sets.

Creating a configuration set

You can use the Amazon SES console or the CreateConfigurationSet API to create new configuration sets.

**To create a configuration set by using the Amazon SES console**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, choose Configuration Sets.
3. Choose Create Configuration Set.
4. For Configuration set name, type a name for the configuration set.

   **Note**
   The name can contain up to 64 alphanumeric characters. It can also contain hyphens (-) and underscores (_). Names can’t contain spaces, accented characters, or any other special characters.

You can also use the CreateConfigurationSet API to create configuration sets. A common way to call this API is by using the AWS CLI.

**To create a configuration set by using the AWS CLI**

- At the command line, type the following command:

  ```bash
  aws ses create-configuration-set --configuration-set Name=ConfigSet
  ```

  In the preceding command, replace ConfigSet with the name that you want to give the configuration set.

   **Note**
   The name can contain up to 64 alphanumeric characters. It can also contain hyphens (-) and underscores (_). Names can’t contain spaces, accented characters, or any other special characters.

For more information about using the CreateConfigurationSet API to create configuration sets, see the Amazon Simple Email Service API Reference.

**Viewing a list of your configuration sets**

You can use the Amazon SES console or you can use the ListConfigurationSets API to view a list of your configuration sets.

**To view your configuration sets using the Amazon SES console**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, choose Configuration Sets.

   In the details pane, you will see a list of your configuration sets.

You can also use the ListConfigurationSets API to view a list of configuration sets. A common way to call this API is by using the AWS CLI.

**To view a list of configuration sets by using the AWS CLI**

- At the command line, type the following command:

  ```bash
  aws ses list-configuration-sets
  ```

For more information about using ListConfigurationSets API to list your configuration sets, see the Amazon Simple Email Service API Reference.
Viewing the details of a configuration set

You can use the Amazon SES console to view the details of a configuration set, or you can use the DescribeConfigurationSet API to describe a configuration set.

Viewing the details of a configuration set using the Amazon SES console

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, choose Configuration Sets.
3. In the details pane, choose the expand icon next to the configuration set.

You will see the details of the configuration set.

You can also use the DescribeConfigurationSet API to show more information about a configuration set. A common way to call this API is by using the AWS CLI.

To obtain more information about a configuration set by using the AWS CLI

- At the command line, type the following command:

  ```
  aws ses describe-configuration-set --configuration-set-name ConfigSet
  ```

  In the preceding command, replace `ConfigSet` with the name of the configuration set that you want to learn more about.

For information about how to use the DescribeConfigurationSet API to describe a configuration set, see the Amazon Simple Email Service API Reference.

Deleting a configuration set

You can use the Amazon SES console or the DeleteConfigurationSet API to delete a configuration set.

To delete a configuration set using the Amazon SES console

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, choose Configuration Sets.
3. In the details pane, choose the configuration set.
4. From the Actions menu, choose Delete, and then confirm that you want to delete the configuration set.

You can also use the DeleteConfigurationSet API to delete configuration sets. A common way to call this API is by using the AWS CLI.

To delete a configuration set by using the AWS CLI

- At the command line, type the following command:

  ```
  aws ses delete-configuration-set --configuration-set ConfigSet
  ```
Managing default configuration sets

In the preceding command, replace `ConfigSet` with the name of the configuration set that you want to delete.

For more information using the `DeleteConfigurationSet` API to delete a configuration set, see the Amazon Simple Email Service API Reference.

Managing Amazon SES default configuration sets

You can add a default configuration set to an email identity when you create the identity, or you can add a default configuration set to an existing identity. You can apply default configuration sets to both email address and domain identities.

A default configuration set automatically applies its rules to all messages that you send from the email identity associated with that configuration set.

Default configuration set considerations

- The configuration set must be created first before associating it with an identity.
- Default configuration sets will only be applied if the identity is verified.
- An email identity can be associated with only one configuration set at a time. However, you can apply the same configuration set to multiple identities.
- A default configuration set at the email address level overrides a default configuration set at the domain level. For example, a default configuration set associated with `joe@example.com` overrides the configuration set for the domain of `example.com`.
- A default configuration set at the domain level applies to all email addresses for that domain (unless you verify specific addresses for the domain).
- If you delete a configuration set that's designated as the default configuration set for an identity, and then attempt to send email through that identity, your call to Amazon SES fails with a "bad request" error.

Creating and verifying an identity with a default configuration set using the Amazon SES API

You can use the `CreateEmailIdentity` operation in the Amazon SES API v2 to create a new email identity and set its default configuration set at the same time.

**Note**
Before you complete the procedure in this section, you have to install and configure the AWS CLI. For more information, see the AWS Command Line Interface User Guide.

To set a default configuration set using the AWS CLI

- At the command line, enter the following command to use the `CreateEmailIdentity` operation.

```bash
aws sesv2 create-email-identity --email-identity ADDRESS-OR-DOMAIN --configuration-set-name CONFIG-SET
```

In the preceding commands, replace `ADDRESS-OR-DOMAIN` with the email identity that you want to verify. Replace `CONFIG-SET` with the name of the configuration set you want to set as the default configuration set for the identity.

If the command executes successfully, it exits without providing any output.
To verify your email address

1. Check the inbox for the email address that you’re verifying. You’ll receive a message with the following subject line: “Amazon Web Services - Email Address Verification Request in region RegionName,” where RegionName is the name of the AWS Region that you attempted to verify the email address in.

Open the message, and then click the link in it.

   Note
   The link in the verification message expires 24 hours after the message was sent. If 24 hours have passed since you received the verification email, repeat steps 1–5 to receive a verification email with a valid link.

2. In the Amazon SES console, under Identity Management, choose Email Addresses. In the list of email addresses, locate the email address you’re verifying. If the email address was verified, the value in the Status column is “verified”.

To verify your domain

To verify your domain, see Verifying a domain with Amazon SES (p. 60) for more information.

Adding a default configuration set to an existing email identity using the Amazon SES API

You can use the PutEmailIdentityConfigurationSetAttributes operation to add or remove a default configuration set from an existing email identity.

   Note
   Before you complete the procedure in this section, you have to install and configure the AWS CLI. For more information, see the AWS Command Line Interface User Guide.

To add a default configuration set using the AWS CLI

- At the command line, enter the following command to use the PutEmailIdentityConfigurationSetAttributes operation.

```
aws sesv2 put-email-identity-configuration-set-attributes --email-identity ADDRESS-OR-DOMAIN --configuration-set-name CONFIG-SET
```

In the preceding commands, replace ADDRESS-OR-DOMAIN with the email identity that you want to verify. Replace CONFIG-SET with the name of the configuration set you wish to set as the identity’s default configuration set.

If the command executes successfully, it exits without providing any output.

To remove a default configuration set using the AWS CLI

- At the command line, enter the following command to use the PutEmailIdentityConfigurationSetAttributes operation.

```
aws sesv2 put-email-identity-configuration-set-attributes --email-identity ADDRESS-OR-DOMAIN
```

In the preceding commands, replace ADDRESS-OR-DOMAIN with the email identity that you want to verify.
If the command executes successfully, it exits without providing any output.

## Returning a default configuration set using the Amazon SES API

You can use the `GetEmailIdentity` operation to return the default configuration set for an email identity, if applicable.

**Note**
Before you complete the procedure in this section, you have to install and configure the AWS CLI. For more information, see the [AWS Command Line Interface User Guide](https://docs.aws.amazon.com/cli/latest/userguide/).

### To return a default configuration set using the AWS CLI

- At the command line, enter the following command to use the `GetEmailIdentity` operation.

```
aws sesv2 get-email-identity --email-identity ADDRESS-OR-DOMAIN
```

In the preceding commands, replace `ADDRESS-OR-DOMAIN` with the email identity for which you wish to know the default configuration set, if any.

If the command executes successfully, it provides a JSON object with the email identity details.

## Overriding a default configuration set using the Amazon SES API

You can use the `SendEmail` operation to send email with a different configuration set. If you do, the configuration set that you specify overrides the default configuration set for the identity.

**Note**
Before you complete the procedure in this section, you have to install and configure the AWS CLI. For more information, see the [AWS Command Line Interface User Guide](https://docs.aws.amazon.com/cli/latest/userguide/).

### To override a default configuration set using the AWS CLI

- At the command line, enter the following command to use the `SendEmail` operation.

```
aws sesv2 send-email --destination file://DESTINATION-JSON --content file://CONTENT-JSON --from-email-address ADDRESS-OR-DOMAIN --configuration-set-name CONFIG-SET
```

In the preceding commands, replace `DESTINATION-JSON` with your destination JSON file, `CONTENT-JSON` with your content JSON file, `ADDRESS-OR-DOMAIN` with your FROM email address, and `CONFIG-SET` with the name of the configuration set you wish to use instead of the default configuration set for the identity.

If the command executes successfully, it outputs a `MessageId`.

## Managing event destinations in Amazon SES

Event destinations allow you to publish email sending metrics—including the numbers of sends, deliveries, opens, clicks, bounces, and complaints—to other AWS products. To learn more about setting up event publishing, see the section called “Monitoring sending using event publishing” (p. 288).
Updating an event destination

You can use the Amazon SES console or the `UpdateConfigurationSetEventDestination` API to update an event destination.

To update an event destination (console)

1. Sign in to the AWS Management Console and open the Amazon SES console at [https://console.aws.amazon.com/ses/](https://console.aws.amazon.com/ses/).
2. In the left navigation pane, choose Configuration Sets.
3. In the configuration set list, choose the configuration set that contains the event destination that you want to update.
4. In the Destination list, to the right of the destination you want to edit, choose the edit icon (edit).
5. Edit the event destination details, and then choose Save.
6. To exit the Edit Configuration Set page, use the back button of your browser.

For information about how to use the `UpdateConfigurationSetEventDestination` API to update an event destination, see the Amazon Simple Email Service API Reference.

Deleting an event destination

You can use the Amazon SES console or the `DeleteConfigurationSetEventDestination` API to delete an event destination.

To delete an event destination (console)

1. Sign in to the AWS Management Console and open the Amazon SES console at [https://console.aws.amazon.com/ses/](https://console.aws.amazon.com/ses/).
2. In the left navigation pane, choose Configuration Sets.
3. In the configuration set list, choose the configuration set that contains the event destination that you want to delete.
4. In the Destination list, choose the delete icon (delete).
5. Confirm that you want to delete the configuration set.
6. To exit the Edit Configuration Set page, use the back button of your browser.

For information about how to use the `DeleteConfigurationSetEventDestination` API to delete an event destination, see the Amazon Simple Email Service API Reference.

Enabling or disabling an event destination

You can use the Amazon SES console or the `UpdateConfigurationSetEventDestination` API to enable or disable an event destination.

To enable or disable an event destination (console)

1. Sign in to the AWS Management Console and open the Amazon SES console at [https://console.aws.amazon.com/ses/](https://console.aws.amazon.com/ses/).
2. In the left navigation pane, choose Configuration Sets.
3. In the configuration set list, choose the configuration set that contains the event destination that you want to enable or disable.
4. In the Destination list, to the right of the destination you want to edit, choose the edit icon (the pencil).
5. Select or deselect Enabled, and then choose Save.
6. To exit the Edit Configuration Set page, use the back button of your browser.

For information about how to use the UpdateConfigurationSetEventDestination API to enable or disable an event destination, see the Amazon Simple Email Service API Reference.

Managing dedicated IP pools

You can use IP pools to create groups of dedicated IP addresses for sending specific types of email. You can also use a pool of IP addresses that are shared by all Amazon SES customers.

Assigning an IP pool to an existing configuration set

You can use the Amazon SES console to associate an IP pool with an existing configuration set.

To assign an IP pool to a configuration set

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, choose Configuration Sets.
3. In the list of configuration sets, choose the configuration set that you want to associate with an IP pool.
4. On the Sending IP pool tab, for Pool name, choose from one of the following options:
   - **A specific dedicated IP pool** – When you select an existing dedicated IP pool, emails that use the configuration set are sent using only the dedicated IP addresses that belong to that pool. For procedures for creating new IP pools, see Creating Dedicated IP Pools (p. 179).
   - **ses-default-dedicated-pool** – This pool contains all of the dedicated IP addresses for your account that do not already belong to an IP pool. If you send an email using a configuration set that is not associated with a pool, or if you send an email without specifying a configuration set at all, the email is sent from one of the addresses in the default pool.
   - **ses-shared-pool** – This pool contains a large set of IP addresses that are shared among all Amazon SES customers. This option may be useful when you need to send email that doesn't align with your usual sending behaviors.

When you are finished, choose Assign.

Modifying IP pool assignments

You can also use the Amazon SES console to assign a different pool to a configuration set that is already associated with a pool. Assigning a different pool to a configuration set overwrites the previous association.

To edit an IP pool assignment

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, choose Configuration Sets.
3. In the list of configuration sets, choose the configuration set that you want to modify.
4. On the Sending IP pool tab, under Assign an IP pool, choose the edit icon (ู่).
5. For Pool name, select the pool that you want to use, and then choose Assign.

**Specifying a configuration set when you send email**

To use a configuration set when sending an email, you must pass the name of the configuration set in the headers of the email. All of the Amazon SES email sending methods—including the AWS CLI, the AWS SDKs, and the Amazon SES SMTP interface (p. 85)—allow you to pass a configuration set in the headers of the email you send.

If you are using the SMTP interface (p. 85) or the SendRawEmail API operation, you can specify a configuration set by including the following header in your email (replacing `ConfigSet` with the name of the configuration set you want to use):

```
X-SES-CONFIGURATION-SET: ConfigSet
```

This guide includes code examples for sending email using Postfix, the AWS SDKs, and the Amazon SES SMTP interface. Each of these examples includes a method of specifying a configuration set. To see step-by-step procedures for sending emails that include references to configuration sets, see the following:

- Integrating Amazon SES with Postfix (p. 93)
- Send an email through Amazon SES using an AWS SDK (p. 30)
- Send an email through Amazon SES using SMTP (p. 20)

You can find additional code examples in the Amazon SES code examples (p. 416) section.

**Exporting reputation metrics for a configuration set to CloudWatch**

Amazon SES automatically exports information about the overall bounce and complaint rates for your entire account to Amazon CloudWatch. You can use these metrics to create alarms in CloudWatch, or to automatically pause email sending using a Lambda function.

You can also export reputation metrics for individual configuration sets to CloudWatch. Exporting reputation data at the configuration set level gives you more control over your sender reputation.

This section includes procedures for exporting reputation data for individual configuration sets to CloudWatch by using the Amazon SES API.

**Enabling the exporting of reputation metrics for a configuration set**

To start exporting reputation metrics for a configuration set, use the UpdateConfigurationSetReputationMetricsEnabled API operation. To access the Amazon SES API, we recommend using the AWS CLI or one of the AWS SDKs.
This procedure assumes that the AWS CLI is installed on your computer and properly configured. For more information about installing and configuring the AWS CLI, see the AWS Command Line Interface User Guide.

To enable the exporting of reputation metrics for a configuration set

- At the command line, type the following command: `aws ses update-configuration-set-reputation-metrics-enabled --configuration-set-name ConfigSet --enabled`

  Replace `ConfigSet` in the preceding command with the name of the configuration set for which you want to start exporting reputation metrics.

Disabling the exporting of reputation metrics for a configuration set

You can also use the `UpdateConfigurationSetReputationMetricsEnabled` API operation to disable the exporting of reputation metrics for a configuration set.

To disable the exporting of reputation metrics for a configuration set

- At the command line, type the following command: `aws ses update-configuration-set-reputation-metrics-enabled --configuration-set-name ConfigSet --no-enabled`

  Replace `ConfigSet` in the preceding command with the name of the configuration set for which you want to disable the exporting of reputation metrics.
Monitoring your Amazon SES sending activity

Amazon SES provides methods to monitor your sending activity. We recommend that you implement these methods so that you can keep track of important measures, such as your account’s bounce, complaint and reject rates. Excessively high bounce and complaint rates may jeopardize your ability to send emails using Amazon SES.

You can also use these methods to measure the rates at which your customers engage with the emails you send. For example, these sending metrics can help you identify your overall open and clickthrough rates.

The metrics that you can measure using Amazon SES are referred to as email sending events. The email sending events that you can monitor are:

- **Sends** – The call to Amazon SES was successful and Amazon SES will attempt to deliver the email.
- **Rejects** – Amazon SES accepted the email, determined that it contained a virus, and rejected it. Amazon SES didn’t attempt to deliver the email to the recipient’s mail server.
- **Bounces** – The recipient’s mail server permanently rejected the email. This event corresponds to hard bounces. Soft bounces are only included when Amazon SES fails to deliver the email after retrying for a period of time.
- **Complaints** – The email was successfully delivered to the recipient. The recipient marked the email as spam.
- **Deliveries** – Amazon SES successfully delivered the email to the recipient’s mail server.
- **Opens** – The recipient received the message and opened it in their email client.
- **Clicks** – The recipient clicked one or more links in the email.
- **Rendering Failures** – The email wasn’t sent because of a template rendering issue. This event type only occurs when you send email using the `SendTemplatedEmail` or `SendBulkTemplatedEmail` API operations. This event type can occur when template data is missing, or when there is a mismatch between template parameters and data.
- **Delivery Delays** – The email couldn’t be delivered to the recipient because a temporary issue occurred. Delivery delays can occur, for example, when the recipient’s inbox is full, or when the receiving email server experiences a transient issue.

You can monitor email sending events in several ways. The method you choose depends on the type of event you want to monitor, the granularity and level of detail you want to monitor it with, and the location where you want Amazon SES to publish the data. You’re required to use either feedback notifications or event publishing to track bounce and complaint events. You can also choose to use multiple monitoring methods. The characteristics of each method are listed in the following table.

<table>
<thead>
<tr>
<th>Monitoring Method</th>
<th>Events You Can Monitor</th>
<th>How to Access the Data</th>
<th>Level of Detail</th>
<th>Granularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon SES console</td>
<td>Deliveries and rejects</td>
<td>Sending Statistics page (p. 261) in Amazon SES console</td>
<td>Count only</td>
<td>Across entire AWS account</td>
</tr>
<tr>
<td>Amazon SES console</td>
<td>Bounce and complaint rates</td>
<td>Reputation Dashboard</td>
<td>Calculated rates only</td>
<td>Across entire AWS account</td>
</tr>
<tr>
<td>Monitoring Method</td>
<td>Events You Can Monitor</td>
<td>How to Access the Data</td>
<td>Level of Detail</td>
<td>Granularity</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------</td>
<td>------------------------</td>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Amazon SES API</td>
<td>Deliveries, bounces, complaints, and rejects</td>
<td>GetSendStatistics API operation</td>
<td>Count only</td>
<td>Across entire AWS account</td>
</tr>
<tr>
<td>Amazon CloudWatch console</td>
<td>Sends, deliveries, opens, clicks, bounces, bounce rate, complaints, complaint rate, and rejects</td>
<td>CloudWatch console</td>
<td>Count only</td>
<td>Across entire AWS account</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note</strong> Some metrics don't appear in CloudWatch until the associated event occurs. For example, bounce metrics don't appear in CloudWatch until at least one email that you send bounces, or until you generate a simulated bounce event by using the mailbox simulator (p. 181).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feedback notifications</td>
<td>Deliveries, bounces, and complaints</td>
<td>Amazon SNS notification (deliveries, bounces, and complaints) or email (bounces and complaints only)</td>
<td>Details on each event</td>
<td>Across entire AWS account</td>
</tr>
</tbody>
</table>
Monitoring your sending statistics using the Amazon SES console

You can monitor the number of emails delivered from your account, as well as the number of messages that have been rejected, directly from the Amazon SES console. The delivery and rejection rates for your account are displayed on the Sending Statistics page.

You can find information about bounces and complaints on the Reputation Dashboard. For more information, see Monitoring your Amazon SES sender reputation (p. 369).

To view delivery and rejection metrics

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, choose Sending Statistics. Your usage statistics are shown under Your Amazon SES Metrics.
3. To view trend data for any metric, double-click the corresponding graph. When you double-click a graph, you can also change the analysis period.

Monitoring your usage statistics using the Amazon SES API

The Amazon SES API provides the GetSendStatistics operation, which returns information about your service usage. We recommend that you check your sending statistics regularly, so that you can make adjustments if needed.
When you call the `GetSendStatistics` operation, you receive a list of data points representing the last two weeks of your sending activity. Each data point in this list represents 15 minutes of activity and contains the following information for that period:

- The number of hard bounces
- The number of complaints
- The number of delivery attempts (corresponds to the number of emails you have sent)
- The number of rejected send attempts
- A timestamp for the analysis period

For a complete description of the `GetSendStatistics` operation, see the Amazon Simple Email Service API Reference.

In this section, you will find the following topics:

- the section called "Calling the `GetSendStatistics` API operation using the AWS CLI" (p. 262)
- the section called “Calling the `GetSendStatistics` operation programmatically” (p. 262)

### Calling the `GetSendStatistics` API operation using the AWS CLI

The easiest way to call the `GetSendStatistics` API operation is to use the AWS Command Line Interface (AWS CLI).

**To call the `GetSendStatistics` API operation using the AWS CLI**

1. If you have not already done so, install the AWS CLI. For more information, see "Installing the AWS Command Line Interface" in the [AWS Command Line Interface User Guide](https://docs.aws.amazon.com/cli/latest/userguide/getting-started.html).
2. If you have not already done so, configure the AWS CLI to use your AWS credentials. For more information, see "Configuring the AWS CLI" in the [AWS Command Line Interface User Guide](https://docs.aws.amazon.com/cli/latest/userguide/cli-configure.html).
3. At the command line, type `aws ses get-send-statistics`

If the AWS CLI is properly configured, you see a list of sending statistics in JSON format. Each JSON object includes aggregated sending statistics for a 15-minute period.

### Calling the `GetSendStatistics` operation programmatically

You can also call the `GetSendStatistics` operation using the AWS SDKs. This section includes code examples for the AWS SDKs for Go, PHP, Python, and Ruby. Choose one of the following links to view code examples for that language:

- Code example for the AWS SDK for Go (p. 263)
- Code example for the AWS SDK for PHP (p. 263)
- Code example for the AWS SDK for Python (Boto) (p. 264)
- Code example for the AWS SDK for Ruby (p. 264)
Calling the GetSendStatistics operation programmatically

Note
These code examples assume that you have created an AWS shared credentials file that contains your AWS Access Key ID, your AWS Secret Access Key, and your preferred AWS Region. For more information, see Create a shared credentials file (p. 31).

Calling GetSendStatistics using the AWS SDK for Go

```go
package main

import (
    "fmt"
    
    // go get github.com/aws/aws-sdk-go/...
    "github.com/aws/aws-sdk-go/aws"
    "github.com/aws/aws-sdk-go/aws/session"
    "github.com/aws/aws-sdk-go/service/ses"
    "github.com/aws/aws-sdk-go/aws/awserr"
)

const (    // Replace us-west-2 with the AWS Region you're using for Amazon SES.
    AwsRegion = "us-west-2"
)

func main() {
    // Create a new session and specify an AWS Region.
    sess, err := session.NewSession(&aws.Config{
        Region:aws.String(AwsRegion)},
    )
    // Create an SES client in the session.
    svc := ses.New(sess)
    input := &ses.GetSendStatisticsInput{}

    result, err := svc.GetSendStatistics(input)
    // Display error messages if they occur.
    if err != nil {
        if aerr, ok := err.(awserr.Error); ok {
            switch aerr.Code() {
            default:
                fmt.Println(aerr.Error())
            }
        } else {
            // Print the error, cast err to awserr.Error to get the Code and
            // Message from an error.
            fmt.Println(err.Error())
        }
    }

    fmt.Println(result)
}
```

Calling GetSendStatistics using the AWS SDK for PHP

```php
<?php
    // Replace path_to_sdk_inclusion with the path to the SDK as described in
    define('REQUIRED_FILE','path_to_sdk_inclusion');
```
Calling the \texttt{GetSendStatistics} operation programmatically

```php
// Replace us-west-2 with the AWS Region you're using for Amazon SES.
define('REGION','us-west-2');

require REQUIRED_FILE;

use Aws\Ses\SesClient;

$client = SesClient::factory(array(
    'version'=> 'latest',
    'region' => REGION
));

try {
    $result = $client->getSendStatistics([]);
    echo($result);
} catch (Exception $e) {
    echo($e->getMessage()."\n");
}
```

Calling \texttt{GetSendStatistics} using the AWS SDK for Python (\texttt{Boto})

```python
import boto3 #pip install boto3
import json
from botocore.exceptions import ClientError

client = boto3.client('ses')

try:
    response = client.get_send_statistics(
)
except ClientError as e:
    print(e.response['Error']['Message'])
else:
    print(json.dumps(response, indent=4, sort_keys=True, default=str))
```

Calling \texttt{GetSendStatistics} using the AWS SDK for Ruby

```ruby
require 'aws-sdk' # gem install aws-sdk
require 'json'

# Replace us-west-2 with the AWS Region you're using for Amazon SES.
awsregion = "us-west-2"

# Create a new SES resource and specify a region
ses = Aws::SES::Client.new(region: awsregion)

begin

    resp = ses.get_send_statistics({
    })
    puts JSON.pretty_generate(resp.to_h)

    # If something goes wrong, display an error message.
    rescue Aws::SES::Errors::ServiceError => error
        puts error
end
```
Monitoring Amazon SES email sending using notifications

In order to send email using Amazon SES, you must have a system in place for managing bounces and complaints. Amazon SES can notify you of bounce or complaint events in three ways: by sending a notification email, by notifying an Amazon SNS topic, or by publishing sending events. This section contains information about setting up Amazon SES to send certain kinds of notifications by email or by notifying an Amazon SNS topic. For more information about publishing sending events, see Monitor email sending using Amazon SES event publishing (p. 288).

You can set up notifications using the Amazon SES console or the Amazon SES API.

Topics

• Important considerations (p. 265)
• Amazon SES notifications sent by email (p. 265)
• Amazon SES notifications sent through Amazon SNS (p. 268)

Important considerations

There are several important points to consider when you set up Amazon SES to send notifications:

• Email and Amazon SNS notifications apply to individual identities (the verified email addresses or domains you use to send email). When you enable notifications for an identity, Amazon SES only sends notifications for emails sent from that identity, and only in the AWS Region you configured notifications in.

• You have to enable one method of receiving bounce or complaint notifications. You can send notifications to the domain or email address that generated the bounce or complaint, or to an Amazon SNS topic. You can also use event publishing (p. 288) to send notifications about several different types of events (including bounces, complaints, deliveries, and more) to an Amazon SNS topic or an Kinesis Data Firehose stream.

If you don't set up one of these methods of receiving bounce or complaint notifications, Amazon SES automatically forwards bounce and complaint notifications to the Return-Path address (or the Source address, if you didn't specify a Return-Path address) in the email that resulted in the bounce or complaint event, even if you disabled email feedback forwarding.

If you disable email feedback forwarding and enable event publishing, you must apply the configuration set that contains the event publishing rule to all emails you send. In this situation, if you don't use the configuration set, Amazon SES automatically forwards bounce and complaint notifications to the Return-Path or Source address in the email that resulted in the bounce or complaint event.

• If you set up Amazon SES to send bounce and complaint events using more than one method (such as by sending email notifications and by using sending events), you may receive more than one notification for the same event.

Amazon SES notifications sent by email

Amazon SES can send you email when you receive bounces and complaints by using a process called email feedback forwarding.

In order to send email using Amazon SES, you must configure it to send bounce and complaint notifications by using one of the following methods:
Notifications through email

- By enabling email feedback forwarding. The procedure for setting up this type of notification is included in this section.
- By sending notifications to an Amazon SNS topic. For more information, see Amazon SES notifications sent through Amazon SNS (p. 268).
- By publishing event notifications. For more information, see Monitor email sending using Amazon SES event publishing (p. 288).

**Important**
For several important points about notifications, see Monitoring Amazon SES email sending using notifications (p. 265).

**Topics**
- Enabling email feedback forwarding (p. 266)
- Disabling email feedback forwarding (p. 266)
- Email feedback forwarding destination (p. 267)

**Enabling email feedback forwarding**

Email feedback forwarding is enabled by default. If you previously disabled it, you can enable it by following the procedures in this section.

**To enable bounce and complaint forwarding through email using the Amazon SES console**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, under **Identity Management**, choose **Email Addresses** if you want to configure bounce and complaint notifications for an email address, or choose **Domains** if you want to configure bounce and complaint notifications for a domain.
3. In the list of verified email addresses or domains, choose the email address or domain that you want to configure bounce and complaint notifications for.
4. In the details pane, expand the **Notifications** section.
5. Choose **Edit Configuration**.
6. Under **Email Feedback Forwarding**, choose **Enabled**.

   **Note**
   Changes you make on this page may take a few minutes to take effect.

You can also enable bounce and complaint notifications through email by using the `SetIdentityFeedbackForwardingEnabled` API operation.

**Disabling email feedback forwarding**

If you set up a different method of providing bounce and complaint notifications, you can disable email feedback forwarding so that you don't receive multiple notifications when a bounce or complaint event occurs.

**To disable bounce and complaint forwarding through email using the Amazon SES console**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, under **Identity Management**, choose **Email Addresses** if you want to configure bounce and complaint notifications for an email address, or choose **Domains** if you want to configure bounce and complaint notifications for a domain.
3. In the list of verified email addresses or domains, choose the email address or domain that you want to configure bounce and complaint notifications for.

4. In the details pane, expand the **Notifications** section.

5. Choose **Edit Configuration**.

6. Under **Email Feedback Forwarding**, choose **Disabled**.

   **Note**
   You must configure one method of receiving bounce and complaint notifications in order to send email through Amazon SES. If you disable email feedback forwarding, you must enable notifications sent by Amazon SNS, or publish bounce and complaint events to an Amazon SNS topic or a Kinesis Data Firehose stream by using [event publishing](p. 288). If you use event publishing, you must also apply the configuration set that contains the event publishing rule to each email you send. If you don't set up a method of receiving bounce and complaint notifications, Amazon SES automatically forwards feedback notifications by email to the address in the Return-Path field (or the Source field, if you didn't specify a Return-Path address) of the message that resulted in the bounce or complaint event. In this situation, Amazon SES forwards bounce and complaint notifications even if you disabled email feedback notifications.

7. Choose **Save Config** to save your notification configuration.

   **Note**
   Changes you make on this page might take a few minutes to take effect.

You can also disable bounce and complaint notifications through email by using the **SetIdentityFeedbackForwardingEnabled** API operation.

### Email feedback forwarding destination

When you receive notifications by email, Amazon SES rewrites the **From** header and sends the notification to you. The address to which Amazon SES forwards the notification depends on how you sent the original message.

If you used the SMTP interface to send the message, then notifications go to the address specified in the MAIL FROM command.

If you used the **SendEmail** API operation to send the message, then the notifications are delivered according to the following rules:

- If you specified the optional **ReturnPath** parameter in your call to the **SendEmail** API, then notifications go to that address.
- Otherwise, notifications go to the address specified in the required **Source** parameter of **SendEmail**.

If you used the **SendRawEmail** API operation to send the message, then the notifications are delivered according to the following rules:

- If you specified a **Source** parameter in your call to the **SendRawEmail** API, then notifications go to that address. This is true even if you specified a Return-Path header in the body of the email.
- Otherwise, if you specified a Return-Path header in the raw message, then notifications go to that address.
- Otherwise, notifications go to the address in the **From** header of the raw message.

   **Note**
   When you specify a Return-Path address in an email, you receive notifications at that address. However, the version of the message that the recipient receives contains a Return-Path header.
that includes an anonymized email address (such as a0b1c2d3e4f5a6b7-c8d9e0f1-a2b3-c4d5-e6f7-a8b9c0d1e2f3-000000@amazonses.com). This anonymization happens regardless of how you sent the email.

Amazon SES notifications sent through Amazon SNS

You can configure Amazon SES to notify an Amazon SNS topic when you receive bounces or complaints, or when emails are delivered. Amazon SNS notifications are in JavaScript Object Notation (JSON) format, which enables you to process them programmatically.

In order to send email using Amazon SES, you must configure it to send bounce and complaint notifications by using one of the following methods:

- By sending notifications to an Amazon SNS topic. The procedure for setting up this type of notification is included in this section.
- By enabling email feedback forwarding. For more information, see Amazon SES notifications sent by email (p. 265).
- By publishing event notifications. For more information, see Monitor email sending using Amazon SES event publishing (p. 288).

Important
See Monitoring Amazon SES email sending using notifications (p. 265) for important information about notifications.

Topics
- Configuring Amazon SNS notifications for Amazon SES (p. 268)
- Amazon SNS notification contents for Amazon SES (p. 271)
- Amazon SNS notification examples for Amazon SES (p. 281)

Configuring Amazon SNS notifications for Amazon SES

Amazon SES can notify you of your bounces, complaints, and deliveries through Amazon Simple Notification Service (Amazon SNS).

You can configure notifications in the Amazon SES console, or by using the Amazon SES API.

Topics in this section:
- Prerequisites (p. 268)
- Configuring notifications using the Amazon SES console (p. 270)
- Configuring notifications using the Amazon SES API (p. 271)
- Troubleshooting feedback notifications (p. 271)

Prerequisites

Complete the following steps before you set up Amazon SNS notifications in Amazon SES:

1. Create a topic in Amazon SNS. For more information, see Create a Topic in the Amazon Simple Notification Service Developer Guide.

Important
When you create your topic using Amazon SNS, for Type, only choose Standard. (SES does not support FIFO type topics.)
Whether you create a new SNS topic or select an existing one, you need to give access to SES to publish notifications to the topic.

To give Amazon SES permission to publish notifications to the topic, on the Edit topic screen in the SNS console, expand Access policy and in the JSON editor, add the following permission policy:

```
{
  "Version": "2012-10-17",
  "Id": "notification-policy",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ses.amazonaws.com"
      },
      "Action": "sns:Publish",
      "Condition": {
        "StringEquals": {
          "AWS:SourceAccount": "111122223333",
        }
      }
    }
  ]
}
```

Make the following changes to the preceding policy example:

- If you're located in a special partition such as China or AWS GovCloud (US), replace `aws` with the corresponding AWS partition, see Amazon Resource Names (ARNs) in the AWS General Reference.
- Replace `us-west-2` with the AWS Region where you created the SNS topic.
- Replace `111122223333` with your AWS account ID.
- Replace `topic_name` with the name of your SNS topic.
- Replace `identity_name` with the verified identity (email address or domain) that you're subscribing to the SNS topic.

2. Subscribe at least one endpoint to the topic. For example, if you want to receive notifications by text message, subscribe an SMS endpoint (that is, a mobile phone number) to the topic. To receive notifications by email, subscribe an email endpoint (an email address) to the topic.

For more information, see Getting Started in the Amazon Simple Notification Service Developer Guide.

3. (Optional) If your Amazon SNS topic uses AWS Key Management Service (AWS KMS) for server-side encryption, you have to add permissions to the AWS KMS key policy. You can add permissions by attaching the following policy to the AWS KMS key policy:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowSESToUseKMSKey",
      "Effect": "Allow",
      "Principal": {
        "Service": "ses.amazonaws.com"
      },
      "Action": [
        "kms:GenerateDataKey",
        "kms:Decrypt"
      ]
    }
  ]
}
```
Configuring notifications using the Amazon SES console

To configure notifications using the Amazon SES console

1. Open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, under Identity Management, choose Domains or Email Addresses.
3. In the list of verified senders, choose the email address or domain that you want to configure notifications for.

   **Important**
   Verified domain notification settings apply to all mail sent from email addresses in that domain except for email addresses that are also verified.

5. Under SNS Topic Configuration, make the following changes to the Amazon SNS topic configuration:
   a. Choose the Amazon SNS topics you want to use to receive notifications. You can publish multiple event type notifications to the same Amazon SNS topic or to different Amazon SNS topics.

      **Important**
      The Amazon SNS topics that you use for bounce, complaint, and delivery notifications have to be in the same AWS Region that use Amazon SES in. Additionally, you have to subscribe one or more endpoints to the topic in order to receive notifications. For example, if you want to have notifications sent to an email address, you have to subscribe an email endpoint to the topic. For more information, see Getting Started in the Amazon Simple Notification Service Developer Guide.

      If you want to use an Amazon SNS topic that you don't own, you must configure your AWS Identity and Access Management (IAM) policy to allow publishing from the Amazon Resource Name (ARN) of the Amazon SNS topic.

   b. If you want the Amazon SNS notifications to contain the original headers of the emails you pass to Amazon SES, choose Include original headers. This option is only available if you've assigned an Amazon SNS topic to the associated notification type. For information about the contents of the original email headers, see the mail object in Amazon SNS notification contents (p. 271).

6. (Optional) If you choose Amazon SNS topics for both bounces and complaints, you can disable email notifications entirely. To disable email notifications for bounces and complaints, under Email Feedback Forwarding, choose Disable. Delivery notifications are available only through Amazon SNS.
7. Choose Save Config. The changes you made to your notification settings might take a few minutes to take effect.

After you configure your settings, you will start receiving bounce, complaint, and/or delivery notifications to your Amazon SNS topic(s). These notifications are in JavaScript Object Notation (JSON) format and follow the structure described in Amazon SNS notification contents (p. 271).

You will be charged standard Amazon SNS rates for bounce, complaint, and delivery notifications. For more information, see the Amazon SNS pricing page.
Note
If an attempt to publish to your Amazon SNS topic fails because the topic has been deleted or your AWS account no longer has permissions to publish to it, Amazon SES removes the configuration for that topic. Additionally, Amazon SES re-enables bounce and complaint email notifications for the identity, and you receive a notification of the change by email. If multiple identities are configured to use the topic, the topic configuration for each identity is changed when each identity experiences a failure to publish to the topic.

Configuring notifications using the Amazon SES API

You can also configure bounce, complaint, and delivery notifications by using the Amazon SES API. Use the following operations to configure notifications:

- SetIdentityNotificationTopic
- SetIdentityFeedbackForwardingEnabled
- GetIdentityNotificationAttributes
- SetIdentityHeadersInNotificationsEnabled

You can use these API actions to write a customized front-end application for notifications. For a complete description of the API actions related to notifications, see the Amazon Simple Email Service API Reference.

Troubleshooting feedback notifications

Not receiving notifications

If you aren't receiving notifications, make sure that you subscribed an endpoint to the topic that the notifications are sent through. When you subscribe an email endpoint to a topic, you receive an email asking you to confirm your subscription. You have to confirm your subscription before you start receiving email notifications. For more information, see Getting Started in the Amazon Simple Notification Service Developer Guide.

InvalidParameterValue error when choosing a topic

If you receive an error stating that an InvalidParameterValue error occurred, check the Amazon SNS topic to see if it's encrypted using AWS KMS. If it is, you have to modify the policy for the AWS KMS key. See Prerequisites (p. 268) for a sample policy.

Amazon SNS notification contents for Amazon SES

Bounce, complaint, and delivery notifications are published to Amazon Simple Notification Service (Amazon SNS) topics in JavaScript Object Notation (JSON) format. The top-level JSON object contains a notificationType string, a mail object, and either a bounce object, a complaint object, or a delivery object.

See the following sections for descriptions of the different types of objects:

- Top-level JSON object (p. 272)
- mail object (p. 272)
- bounce object (p. 275)
- complaint object (p. 279)
- delivery object (p. 281)

The following are some important notes about the contents of Amazon SNS notifications for Amazon SES:
For a given notification type, you might receive one Amazon SNS notification for multiple recipients, or you might receive a single Amazon SNS notification per recipient. Your code should be able to parse the Amazon SNS notification and handle both cases; Amazon SES does not make ordering or batching guarantees for notifications sent through Amazon SNS. However, different Amazon SNS notification types (for example, bounces and complaints) are never combined into a single notification.

You might receive multiple types of Amazon SNS notifications for one recipient. For example, the receiving mail server might accept the email (triggering a delivery notification), but after processing the email, the receiving mail server might determine that the email actually results in a bounce (triggering a bounce notification). However, these are always separate notifications because they are different notification types.

Amazon SES reserves the right to add additional fields to the notifications. As such, applications that parse these notifications must be flexible enough to handle unknown fields.

Amazon SES overwrites the headers of the message when it sends the email. You can retrieve the headers of the original message from the headers and commonHeaders fields of the mail object.

### Top-Level JSON object

The top-level JSON object in an Amazon SES notification contains the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notificationType</td>
<td>A string that holds the type of notification represented by the JSON object. Possible values are Bounce, Complaint, or Delivery.</td>
</tr>
<tr>
<td>mail</td>
<td>A JSON object that contains information about the original mail to which the notification pertains. For more information, see Mail object (p. 272).</td>
</tr>
<tr>
<td>bounce</td>
<td>This field is present only if the notificationType is Bounce and contains a JSON object that holds information about the bounce. For more information, see Bounce object (p. 275).</td>
</tr>
<tr>
<td>complaint</td>
<td>This field is present only if the notificationType is Complaint and contains a JSON object that holds information about the complaint. For more information, see Complaint object (p. 279).</td>
</tr>
<tr>
<td>delivery</td>
<td>This field is present only if the notificationType is Delivery and contains a JSON object that holds information about the delivery. For more information, see Delivery object (p. 281).</td>
</tr>
</tbody>
</table>

### Mail object

Each bounce, complaint, or delivery notification contains information about the original email in the mail object. The JSON object that contains information about a mail object has the following fields.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timestamp</td>
<td>The time at which the original message was sent (in ISO8601 format).</td>
</tr>
<tr>
<td>messageId</td>
<td>A unique ID that Amazon SES assigned to the message. Amazon SES returned this value to you when you sent the message.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> This message ID was assigned by Amazon SES. You can find the message ID of the original email in the headers and commonHeaders fields of the mail object.</td>
</tr>
<tr>
<td>source</td>
<td>The email address from which the original message was sent (the envelope MAIL FROM address).</td>
</tr>
<tr>
<td>sourceArn</td>
<td>The Amazon Resource Name (ARN) of the identity that was used to send the email. In the case of sending authorization, the sourceArn is the ARN of the identity that the identity owner authorized the delegate sender to use to send the email. For more information about sending authorization, see Using sending authorization (p. 149).</td>
</tr>
<tr>
<td>sourceIp</td>
<td>The originating public IP address of the client that performed the email sending request to Amazon SES.</td>
</tr>
<tr>
<td>sendingAccountId</td>
<td>The AWS account ID of the account that was used to send the email. In the case of sending authorization, the sendingAccountId is the delegate sender's account ID.</td>
</tr>
<tr>
<td>destination</td>
<td>A list of email addresses that were recipients of the original mail.</td>
</tr>
<tr>
<td>headersTruncated</td>
<td>This object is only present if you configured the notification settings to include the headers from the original email. Indicates whether the headers are truncated in the notification. Amazon SES truncates the headers in the notification when the headers from the original message are 10KB or larger in size. Possible values are <code>true</code> and <code>false</code>.</td>
</tr>
<tr>
<td>headers</td>
<td>This object is only present if you configured the notification settings to include the headers from the original email. A list of the email's original headers. Each header in the list has a <code>name</code> field and a <code>value</code> field. <strong>Note</strong> Any message ID within the headers object is from the original message that</td>
</tr>
</tbody>
</table>
### Field Name | Description
--- | ---
messageId | you passed to Amazon SES. The message ID that Amazon SES subsequently assigned to the message is in the `messageId` field of the `mail` object.

**commonHeaders** | This object is only present if you configured the notification settings to include the headers from the original email.

Includes information about common email headers from the original email, including the From, To, and Subject fields. Within this object, each header is a key. The From and To fields are represented by arrays that can contain multiple values.

**Note**
Any message ID within the `commonHeaders` object is from the original message that you passed to Amazon SES. The message ID that Amazon SES subsequently assigned to the message is in the `messageId` field of the `mail` object.

The following is an example of a `mail` object that includes the original email headers. When this notification type is not configured to include the original email headers, the `mail` object does not include the `headersTruncated`, `headers`, and `commonHeaders` fields.

```json
{
    "timestamp":"2018-10-08T14:05:45 +0000",
    "messageId":"000001378603177f-7a5433e7-8edb-42ae-af10-f0181f34d6ee-000000",
    "source":"sender@example.com",
    "sourceIp": "127.0.3.0",
    "sendingAccountId":"123456789012",
    "destination": ["recipient@example.com"],
    "headersTruncated":false,
    "headers": [
        {
            "name":"From",
            "value":""Sender Name" <sender@example.com>"
        },
        {
            "name":"To",
            "value":""Recipient Name" <recipient@example.com>"
        },
        {
            "name":"Message-ID",
            "value":"custom-message-ID"
        },
        {
            "name":"Subject",
            "value":"Hello"
        },
        {
            "name":"Content-Type",
```
Bounce object

The JSON object that contains information about bounces contains the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bounceType</td>
<td>The type of bounce, as determined by Amazon SES. For more information, see Bounce types (p. 277).</td>
</tr>
<tr>
<td>bounceSubType</td>
<td>The subtype of the bounce, as determined by Amazon SES. For more information, see Bounce types (p. 277).</td>
</tr>
<tr>
<td>bouncedRecipients</td>
<td>A list that contains information about the recipients of the original mail that bounced. For more information, see Bounced recipients (p. 276).</td>
</tr>
<tr>
<td>timestamp</td>
<td>The date and time at which the bounce was sent (in ISO8601 format). Note that this is the time at which the notification was sent by the ISP, and not the time at which it was received by Amazon SES.</td>
</tr>
<tr>
<td>feedbackId</td>
<td>A unique ID for the bounce.</td>
</tr>
</tbody>
</table>

If Amazon SES was able to contact the remote Message Transfer Authority (MTA), the following field is also present.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>remoteMtaIp</td>
<td>The IP address of the MTA to which Amazon SES attempted to deliver the email.</td>
</tr>
</tbody>
</table>
If a delivery status notification (DSN) was attached to the bounce, the following field is also present.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reportingMTA</td>
<td>The value of the Reporting-MTA field from the DSN. This is the value of the MTA that attempted to perform the delivery, relay, or gateway operation described in the DSN.</td>
</tr>
</tbody>
</table>

The following is an example of a bounce object.

```json
{
  "bounceType": "Permanent",
  "bounceSubType": "General",
  "bouncedRecipients": [
    {
      "status": "5.0.0",
      "action": "failed",
      "diagnosticCode": "smtp; 550 user unknown",
      "emailAddress": "recipient1@example.com"
    },
    {
      "status": "4.0.0",
      "action": "delayed",
      "emailAddress": "recipient2@example.com"
    }
  ],
  "reportingMTA": "example.com",
  "timestamp": "2012-05-25T14:59:38.605Z",
  "feedbackId": "000001378603176d-5a4b5ad9-6f30-4198-a8c3-b1eb0c270a1d-000000",
  "remoteMtaIp": "127.0.2.0"
}
```

**Bounced recipients**

A bounce notification may pertain to a single recipient or to multiple recipients. The `bouncedRecipients` field holds a list of objects—one per recipient to whom the bounce notification pertains—and always contains the following field.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>emailAddress</td>
<td>The email address of the recipient. If a DSN is available, this is the value of the Final-Recipient field from the DSN.</td>
</tr>
</tbody>
</table>

Optionally, if a DSN is attached to the bounce, the following fields may also be present.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>The value of the Action field from the DSN. This indicates the action performed by the Reporting-MTA as a result of its attempt to deliver the message to this recipient.</td>
</tr>
<tr>
<td>status</td>
<td>The value of the Status field from the DSN. This is the per-recipient transport-independent...</td>
</tr>
<tr>
<td>Field Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>status code</td>
<td>status code that indicates the delivery status of the message.</td>
</tr>
<tr>
<td>diagnosticCode</td>
<td>The status code issued by the reporting MTA. This is the value of the Diagnostic-Code field from the DSN. This field may be absent in the DSN (and therefore also absent in the JSON).</td>
</tr>
</tbody>
</table>

The following is an example of an object that might be in the bouncedRecipients list.

```json
{
    "emailAddress": "recipient@example.com",
    "action": "failed",
    "status": "5.0.0",
    "diagnosticCode": "X-Postfix; unknown user"
}
```

**Bounce types**

The bounce object contains a bounce type of Undetermined, Permanent, or Transient. The Permanent and Transient bounce types can also contain one of several bounce subtypes.

When you receive a bounce notification with a bounce type of Transient, you might be able to send email to that recipient in the future if the issue that caused the message to bounce is resolved.

When you receive a bounce notification with a bounce type of Permanent, it's unlikely that you'll be able to send email to that recipient in the future. For this reason, you should immediately remove the recipient whose address produced the bounce from your mailing lists.

**Note**

When a soft bounce (a bounce related to a temporary issue, such as the recipient's inbox being full) occurs, Amazon SES attempts to redeliver the email for a certain period of time. At the end of that period of time, if Amazon SES still can't deliver the email, it stops trying.

Amazon SES provides notifications for hard bounces, as well as for soft bounces that it stopped trying to deliver. If you want to receive a notification each time a soft bounce occurs, enable event publishing (p. 290) and configure it to send notifications when Delivery Delay events occur.

<table>
<thead>
<tr>
<th>bounceType</th>
<th>bounceSubType</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Undetermined</td>
<td>Undetermined</td>
<td>The recipient's email provider sent a bounce message. The bounce message didn't contain enough information for Amazon SES to determine the reason for the bounce. The bounce email, which was sent to the address in the Return-Path header of the email that resulted in the bounce, might contain additional information about the issue that caused the email to bounce.</td>
</tr>
<tr>
<td>Permanent</td>
<td>General</td>
<td>The recipient's email provider sent a hard bounce message, but didn't specify the reason for the hard bounce.</td>
</tr>
</tbody>
</table>

**Important**

When you receive this type of bounce notification, you should immediately remove the recipient's email address from...
<table>
<thead>
<tr>
<th>bounceType</th>
<th>bounceSubType</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Permanent</td>
<td>NoEmail</td>
<td>The intended recipient's email provider sent a bounce message indicating that the email address doesn't exist.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Important</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>When you receive this type of bounce notification, you should immediately remove the recipient's email address from your mailing list. Sending</td>
</tr>
<tr>
<td></td>
<td></td>
<td>messages to addresses that don't exist can have a negative impact on your reputation as a sender. If you continue sending email to addresses that</td>
</tr>
<tr>
<td></td>
<td></td>
<td>don't exist, we might pause your ability to send additional email.</td>
</tr>
<tr>
<td>Permanent</td>
<td>Suppressed</td>
<td>The recipient's email address is on the Amazon SES suppression list because it has a recent history of producing hard bounces. For information</td>
</tr>
<tr>
<td></td>
<td></td>
<td>about removing an address from the Amazon SES suppression list, see Using the Amazon SES global suppression list (p. 192).</td>
</tr>
<tr>
<td>Permanent</td>
<td>OnAccountSuppressionList</td>
<td>Amazon SES has suppressed sending to this address because it is on the account-level suppression list (p. 184).</td>
</tr>
<tr>
<td>Transient</td>
<td>General</td>
<td>The recipient's email provider sent a general bounce message. You might be able to send a message to the same recipient in the future if the issue</td>
</tr>
<tr>
<td></td>
<td></td>
<td>that caused the message to bounce is resolved.</td>
</tr>
<tr>
<td></td>
<td>MailboxFull</td>
<td>The recipient's email provider sent a bounce message because the recipient's inbox was full. You might be able to send to the same recipient in</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the future when the mailbox is no longer full.</td>
</tr>
</tbody>
</table>
### bounceType | bounceSubType | Description
--- | --- | ---
Transient | MessageTooLarge | The recipient's email provider sent a bounce message because message you sent was too large. You might be able to send a message to the same recipient if you reduce the size of the message.

Transient | ContentRejected | The recipient's email provider sent a bounce message because the message you sent contains content that the provider doesn't allow. You might be able to send a message to the same recipient if you change the content of the message.

Transient | AttachmentRejected | The recipient's email provider sent a bounce message because the message contained an unacceptable attachment. For example, some email providers may reject messages with attachments of a certain file type, or messages with very large attachments. You might be able to send a message to the same recipient if you remove or change the content of the attachment.

### Complaint object

The JSON object that contains information about complaints has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>complainedRecipients</td>
<td>A list that contains information about recipients that may have been responsible for the complaint. For more information, see Complained Recipients (p. 280).</td>
</tr>
<tr>
<td>timestamp</td>
<td>The date and time when the ISP sent the complaint notification, in ISO 8601 format. The date and time in this field might not be the same as the date and time when Amazon SES received the notification.</td>
</tr>
<tr>
<td>feedbackId</td>
<td>A unique ID associated with the complaint.</td>
</tr>
<tr>
<td>complaintSubType</td>
<td>The value of the complaintSubType field can either be null or OnAccountSuppressionList. If the value is OnAccountSuppressionList, Amazon SES accepted the message, but didn't attempt to send it because it was on the account-level suppression list (p. 184).</td>
</tr>
</tbody>
</table>

Further, if a feedback report is attached to the complaint, the following fields may be present.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userAgent</td>
<td>The value of the User-Agent field from the feedback report. This indicates the name and version of the system that generated the report.</td>
</tr>
</tbody>
</table>
The following is an example of a `complaint` object.

```json
{
  "userAgent": "ExampleCorp Feedback Loop (V0.01)",
  "complainedRecipients": [
    {
      "emailAddress": "recipient1@example.com"
    }
  ],
  "complaintFeedbackType": "abuse",
  "arrivalDate": "2009-12-03T04:24:21.000-05:00",
  "timestamp": "2012-05-25T14:59:38.623Z",
  "feedbackId": "000001378603177f-18c07c78-fa81-4a58-9dd1-fedc3cb8f49a-000000"
}
```

### Complained Recipients

The `complainedRecipients` field contains a list of recipients that may have submitted the complaint. You should use this information to determine which recipient submitted the complaint, and then immediately remove that recipient from your mailing lists.

**Important**

Most ISPs remove the email address of the recipient who submitted the complaint from their complaint notification. For this reason, this list contains information about recipients who might have sent the complaint, based on the recipients of the original message and the ISP from which we received the complaint. Amazon SES performs a lookup against the original message to determine this recipient list.

JSON objects in this list contain the following field.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>emailAddress</td>
<td>The email address of the recipient.</td>
</tr>
</tbody>
</table>

The following is an example of a `Complained Recipient` object.

```json
{ "emailAddress": "recipient1@example.com" }
```

**Note**

Because of this behavior, you can be more certain that you know which email address complained about your message if you limit your sending to one message per recipient (rather than sending one message with 30 different email addresses in the bcc line).
Complaint Types

You may see the following complaint types in the complaintFeedbackType field as assigned by the reporting ISP, according to the Internet Assigned Numbers Authority website:

- abuse—Indicates unsolicited email or some other kind of email abuse.
- auth-failure—Email authentication failure report.
- fraud—Indicates some kind of fraud or phishing activity.
- not-spam—Indicates that the entity providing the report does not consider the message to be spam. This may be used to correct a message that was incorrectly tagged or categorized as spam.
- other—Indicates any other feedback that does not fit into other registered types.
- virus—Reports that a virus is found in the originating message.

Delivery object

The JSON object that contains information about deliveries always has the following fields:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timestamp</td>
<td>The time Amazon SES delivered the email to the recipient's mail server (in ISO8601 format).</td>
</tr>
<tr>
<td>processingTimeMillis</td>
<td>The time in milliseconds between when Amazon SES accepted the request from the sender to passing the message to the recipient's mail server.</td>
</tr>
<tr>
<td>recipients</td>
<td>A list of the intended recipients of the email to which the delivery notification applies.</td>
</tr>
<tr>
<td>smtpResponse</td>
<td>The SMTP response message of the remote ISP that accepted the email from Amazon SES. This message varies by email, by receiving mail server, and by receiving ISP.</td>
</tr>
<tr>
<td>reportingMTA</td>
<td>The host name of the Amazon SES mail server that sent the mail.</td>
</tr>
<tr>
<td>remoteMtaIp</td>
<td>The IP address of the MTA to which Amazon SES delivered the email.</td>
</tr>
</tbody>
</table>

The following is an example of a delivery object.

```json
{
    "processingTimeMillis": 546,
    "recipients": ["success@simulator.amazonses.com"],
    "smtpResponse": "250 ok: Message 64111812 accepted",
    "reportingMTA": "a8-70.smtp-out.amazonses.com",
    "remoteMtaIp": "127.0.0.1"
}
```

Amazon SNS notification examples for Amazon SES

The following sections provide examples of the three types of notifications:
• For bounce notification examples, see Amazon SNS Bounce Notification Examples (p. 282).
• For complaint notification examples, see Amazon SNS Complaint Notification Examples (p. 284).
• For delivery notification examples, see Amazon SNS Delivery Notification Example (p. 287).

Amazon SNS Bounce Notification Examples

This section contains examples of bounce notifications with and without a Delivery Status Notification (DSN) provided by the email receiver that sent the feedback.

Bounce Notification With a DSN

The following is an example of a bounce notification that contains a DSN and the original email headers. When bounce notifications are not configured to include the original email headers, the mail object within the notifications does not include the headersTruncated, headers, and commonHeaders fields.

```json
{
    "notificationType": "Bounce",
    "bounce": {
        "bounceType": "Permanent",
        "reportingMTA": "dns; email.example.com",
        "bouncedRecipients": [
            {
                "emailAddress": "jane@example.com",
                "status": "5.1.1",
                "action": "failed",
                "diagnosticCode": "smtp; 550 5.1.1 <jane@example.com>... User"
            }
        ],
        "bounceSubType": "General",
        "timestamp": "2016-01-27T14:59:38.237Z",
        "feedbackId": "00000138111222aa-33322211-cccc-cccc-cccc-ddddaaaa068a-000000",
        "remoteMtaIp": "127.0.2.0"
    },
    "mail": {
        "timestamp": "2016-01-27T14:59:38.237Z",
        "source": "john@example.com",
        "sourceIp": "127.0.3.0",
        "sendingAccountId": "123456789012",
        "messageId": "00000138111222aa-33322211-cccc-cccc-cccc-ddddaaaa0680-000000",
        "destination": [
            "jane@example.com",
            "mary@example.com",
            "richard@example.com"
        ],
        "headersTruncated": false,
        "headers": [
            {
                "name": "From",
                "value": "\"John Doe\" <john@example.com>"
            },
            {
                "name": "To",
                "value": "\"Jane Doe\" <jane@example.com>, \"Mary Doe\" <mary@example.com>, "Richard Doe\" <richard@example.com>"
            },
            {
                "name": "Message-ID",
                "value": "custom-message-ID"
            }
        ]
    }
}
```
Bounce Notification Without a DSN

The following is an example of a bounce notification that includes the original email headers but does not include a DSN. When bounce notifications are not configured to include the original email headers, the `mail` object within the notifications does not include the `headersTruncated`, `headers`, and `commonHeaders` fields.

```json
{
  "notificationType": "Bounce",
  "bounce": {
    "bounceType": "Permanent",
    "bounceSubType": "General",
    "bouncedRecipients": [
      {
        "emailAddress": "jane@example.com"
      },
      {
        "emailAddress": "richard@example.com"
      }
    ],
    "timestamp": "2016-01-27T14:59:38.237Z",
    "feedbackId": "00000137860315fd-869464a4-8680-4114-98d3-716fe35851f9-000000",
    "remoteMtaIp": "127.0.2.0"
  },
  "mail": {
    "timestamp": "2016-01-27T14:59:38.237Z",
    "messageId": "00000137860315fd-34208509-5b74-41f3-95c5-22c1edc3c924-000000",
    "source": "john@example.com",
    "sourceIp": "127.0.2.0",
    "sendingAccountId": "123456789012",
    "destination": [
      "jane@example.com",
      "jane@example.com",
      "jane@example.com"
    ]
  }
}
```
Amazon SNS Complaint Notification Examples

This section contains examples of complaint notifications with and without a feedback report provided by the email receiver that sent the feedback.

Complaint Notification With a Feedback Report

The following is an example of a complaint notification that contains a feedback report and the original email headers. When complaint notifications are not configured to include the original email headers, the mail object within the notifications does not include the headersTruncated, headers, and commonHeaders fields.

```json
{
    "commonHeaders": {
        "from": ["John Doe <john@example.com>"],
        "date": "Wed, 27 Jan 2016 14:05:45 +0000",
        "to": ["Jane Doe <jane@example.com>, Mary Doe <mary@example.com>, Richard Doe <richard@example.com>"],
        "messageId": "custom-message-ID",
        "subject": "Hello"
    }
}
```
"notificationType":"Complaint",
"complaint":{
  "userAgent":"AnyCompany Feedback Loop (V0.01)",
  "complainedRecipients":[
    {
      "emailAddress":"richard@example.com"
    }
  ],
  "complaintFeedbackType":"abuse",
  "arrivalDate":"2016-01-27T14:59:38.237Z",
  "timestamp":"2016-01-27T14:59:38.237Z",
  "feedbackId":"000001378603177f-18c07c78-fa81-4a58-9dd1-fedc3cb8f49a-000000"
},
"mail":{
  "timestamp":"2016-01-27T14:59:38.237Z",
  "messageId":"000001378603177f-7a5433e7-8ed3-42ae-af10-f0181f34d6ee-000000",
  "source":"john@example.com",
  "sourceIp": "127.0.3.0",
  "sendingAccountId":"123456789012",
  "destination":[
    "jane@example.com",
    "mary@example.com",
    "richard@example.com"
  ],
  "headersTruncated":false,
  "headers":{
    "name":"From",
    "value":"\"John Doe\" <john@example.com>"
  },
  "name":"To",
  "value":"\"Jane Doe\" <jane@example.com>, \"Mary Doe\" <mary@example.com>, \"Richard Doe\" <richard@example.com>"
  }},
  "name":"Message-ID",
  "value":"custom-message-ID"
},
  "name":"Subject",
  "value":"Hello"
},
  "name":"Content-Type",
  "value":"text/plain; charset=UTF-8"
},
  "name":"Content-Transfer-Encoding",
  "value":"base64"
},
  "name":"Date",
  "value":"Wed, 27 Jan 2016 14:05:45 +0000"
},
"commonHeaders":{
  "from":[
    "John Doe <john@example.com>"
  ],
  "date":"Wed, 27 Jan 2016 14:05:45 +0000",
  "to":[
    "Jane Doe <jane@example.com>, Mary Doe <mary@example.com>, Richard Doe <richard@example.com>"
  ]}
Complaint Notification Without a Feedback Report

The following is an example of a complaint notification that includes the original email headers but does not include a feedback report. When complaint notifications are not configured to include the original email headers, the `mail` object within the notifications does not include the `headersTruncated`, `headers`, and `commonHeaders` fields.

```json
{
  "notificationType": "Complaint",
  "complaint": {
    "complainedRecipients": [
      {
        "emailAddress": "richard@example.com"
      }
    ],
    "timestamp": "2016-01-27T14:59:38.237Z",
    "feedbackId": "0000013786031775-fea503bc-7497-49e1-881b-a0379bb037d3-000000"
  },
  "mail": {
    "timestamp": "2016-01-27T14:59:38.237Z",
    "messageId": "0000013786031775-163e3910-53eb-4c8e-a04a-f29debf88a84-000000",
    "source": "john@example.com",
    "sourceIp": "127.0.0.1",
    "sendingAccountId": "123456789012",
    "destination": [
      "jane@example.com",
      "mary@example.com",
      "richard@example.com"
    ],
    "headersTruncated": false,
    "headers": [
      {
        "name": "From",
        "value": ""John Doe" <john@example.com>"
      },
      {
        "name": "To",
        "value": ""Jane Doe" <jane@example.com>, "Mary Doe" <mary@example.com>, "Richard Doe" <richard@example.com>"
      },
      {
        "name": "Message-ID",
        "value": "custom-message-ID"
      },
      {
        "name": "Subject",
        "value": "Hello"
      },
      {
        "name": "Content-Type",
        "value": "text/plain; charset="UTF-8""
      },
      {
        "name": "Content-Transfer-Encoding",
        "value": "base64"
      }
    ]
  }
}
```
Amazon SNS Delivery Notification Example

The following is an example of a delivery notification that includes the original email headers. When delivery notifications are not configured to include the original email headers, the mail object within the notifications does not include the headersTruncated, headers, and commonHeaders fields.

```json
{
  "notificationType": "Delivery",
  "mail": {
    "timestamp": "2016-01-27T14:59:38.237Z",
    "messageId": "0000014644fe5ef6-9a483358-9170-4cb4-a269-f5dcdf415321-000000",
    "source": "john@example.com",
    "sourceIp": "127.0.3.0",
    "sendingAccountId": "123456789012",
    "destination": [
      "jane@example.com"
    ],
    "headersTruncated": false,
    "headers": [
      {
        "name": "From",
        "value": ""John Doe" <john@example.com>"
      },
      {
        "name": "To",
        "value": ""Jane Doe" <jane@example.com>"
      },
      {
        "name": "Message-ID",
        "value": "custom-message-ID"
      },
      {
        "name": "Subject",
        "value": "Hello"
      },
      {
        "name": "Content-Type",
        "value": "text/plain; charset="UTF-8""
      },
      {
        "name": "Content-Transfer-Encoding",
        "value": "base64"
      }
    ]
  }
}
```
Monitor email sending using Amazon SES event publishing

To enable you to track your email sending at a granular level, you can set up Amazon SES to publish email sending events to Amazon CloudWatch, Amazon Kinesis Data Firehose, or Amazon Simple Notification Service based on characteristics that you define.

You can track several types of email sending events, including sends, deliveries, opens, clicks, bounces, complaints, rejections, rendering failures, and delivery delays. This information can be useful for operational and analytical purposes. For example, you can publish your email sending data to CloudWatch and create dashboards that track the performance of your email campaigns, or you can use Amazon SNS to send you notifications when certain events occur.

How event publishing works

To use event publishing, you first set up one or more configuration sets. A configuration set specifies where to publish your events and which events to publish. Then, each time you send an email, you provide the name of the configuration set and one or more message tags, in the form of name/value pairs, to categorize the email. For example, if you advertise books, you could name a message tag genre, and assign a value of sci-fi or western, when you send an email for the associated campaign. Depending on which email sending interface you use, you either provide the message tag as a parameter to the API call or as an Amazon SES-specific email header. For more information about configuration sets, see Using Amazon SES configuration sets (p. 249).

In addition to the message tags that you specify, Amazon SES also adds auto-tags to the messages you send. You do not need to perform any additional steps to use auto-tags.

The following table lists the auto-tags that are automatically applied to messages you send using Amazon SES.
How to use event publishing

The following sections contain the information you need to set up and use Amazon SES event publishing.

- Setting up event publishing (p. 290)
- Working with event data (p. 299)
- Tutorials (p. 341)

Event publishing terminology

The following list defines terms related to Amazon SES event publishing.

Email sending event

Information associated with the outcome of an email you submit to Amazon SES. Sending events include the following:

- **Sends** – The call to Amazon SES was successful and Amazon SES will attempt to deliver the email.
- **Rejects** – Amazon SES accepted the email, determined that it contained a virus, and rejected it. Amazon SES didn't attempt to deliver the email to the recipient's mail server.
- **Bounces** – The recipient's mail server permanently rejected the email. This event corresponds to hard bounces. Soft bounces are only included when Amazon SES fails to deliver the email after retrying for a period of time.
- **Complaints** – The email was successfully delivered to the recipient. The recipient marked the email as spam.
- **Deliveries** – Amazon SES successfully delivered the email to the recipient's mail server.
- **Opens** – The recipient received the message and opened it in their email client.
- **Clicks** – The recipient clicked one or more links in the email.
- **Rendering Failures** – The email wasn't sent because of a template rendering issue. This event type only occurs when you send email using the SendTemplatedEmail or SendBulkTemplatedEmail API operations. This event type can occur when template data is missing, or when there is a mismatch between template parameters and data.
- **Delivery Delays** – The email couldn't be delivered to the recipient because a temporary issue occurred. Delivery delays can occur, for example, when the recipient's inbox is full, or when the receiving email server experiences a transient issue.
Configuration set

A set of rules that defines the destination that Amazon SES publishes email sending events to, and the types of email sending events that you want to publish. When you send an email that you want to use with event publishing, you specify the configuration set to associate with the email.

Event destination

An AWS service that you publish Amazon SES email sending events to. Each event destination that you set up belongs to one, and only one, configuration set.

Message tag

A name/value pair that you use to categorize an email for the purpose of event publishing. Examples are campaign/book and campaign/clothing. When you send an email, you either specify the message tag as a parameter to the API call or as an Amazon SES-specific email header.

Auto-tag

Message tags that are automatically included in event publishing reports. There is an auto-tag for the configuration set name, the domain of the "From" address, the caller's outgoing IP address, the Amazon SES outgoing IP address, and the IAM identity of the caller.

Setting up Amazon SES event publishing

This section describes what you need to do to configure Amazon SES to publish your email sending events to Amazon CloudWatch, Amazon Kinesis Data Firehose, or Amazon Simple Notification Service (Amazon SNS).

First, you create a configuration set using the Amazon SES console or API. After you create a configuration set, you add one or more event destinations (CloudWatch, Kinesis Data Firehose, or Amazon SNS) to the configuration set, and configure parameters unique to the event destination. Then, each time you send an email, you specify which configuration set to use.

Topics in this section

- Step 1: Create a configuration set (p. 290)
- Step 2: Add an event destination (p. 291)
- Step 3: Specify your configuration set when you send email (p. 298)

Step 1: Create a configuration set

You can use configuration sets to publish email sending events (bounces, complaints, deliveries, sent emails, rejected emails, rendering failures, and delivery delays) to Amazon CloudWatch or Amazon Kinesis Data Firehose.

You can also create configuration sets using the Amazon SES console or the CreateConfigurationSet operation in the Amazon SES API V2.

To create a configuration set by using the Amazon SES console

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, choose Configuration Sets.
3. In the content pane, choose Create Configuration Set.
4. Type a name for the configuration set, and then choose Create Configuration Set.
5. Choose Close.
Step 2: Add an event destination

Event destinations are places that you publish Amazon SES events to. Each event destination that you set up belongs to one, and only one, configuration set. When you set up an event destination with Amazon SES, you choose the AWS service destination, and you specify parameters associated with that destination.

When you set up an event destination, you can choose to send events to one of the following AWS services: Amazon CloudWatch, Amazon Kinesis Data Firehose, and Amazon Simple Notification Service (Amazon SNS). The event destination that you choose depends on the level of detail you want about the events, and the way you want to receive the event information. If you simply want a running total of each type of event (for example, so that you can set an alarm when the total gets too high), you can use CloudWatch. If you want detailed event records that you can output to another service such as Amazon OpenSearch Service or Amazon Redshift for analysis, you can use Kinesis Data Firehose. If you want to receive notifications when certain events occur, you can use Amazon SNS.

This section contains the following topics

- Set up a CloudWatch event destination for event publishing (p. 291)
- Set up a Kinesis Data Firehose event destination for Amazon SES event publishing (p. 293)
- Set up an Amazon SNS event destination for event publishing (p. 296)

Set up a CloudWatch event destination for event publishing

You can use Amazon CloudWatch event destinations to publish Amazon SES email sending events to CloudWatch. Because a CloudWatch event destination exists within a configuration set only, you must first create a configuration set (p. 290) and then add the event destination to the configuration set.

When you add a CloudWatch event destination to a configuration set, you must choose one or more CloudWatch dimensions that correspond to the message tags you use when you send your emails. Like message tags, a CloudWatch dimension is a name/value pair that helps you uniquely identify a metric.

For example, you might have a message tag and a dimension called campaign that you use to identify your email campaign. When you publish your email sending events to CloudWatch, choosing your message tags and dimensions is important because these choices affect your CloudWatch billing and determine how you can filter your email sending event data in CloudWatch.

This section provides information to help you choose your dimensions, and then shows how to add a CloudWatch event destination to a configuration set.

Topics in this section

- Adding a CloudWatch Event Destination (p. 291)
- Choosing CloudWatch Dimensions (p. 293)

Adding a CloudWatch Event Destination

The procedure in this section shows how to add a CloudWatch event destination to a configuration set.

You can also use the UpdateConfigurationSetEventDestination operation in the Amazon SES API V2 to create and modify event destinations.

To add a CloudWatch event destination to a configuration set by using the console

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, choose Configuration Sets.
3. In the list of configuration sets, choose the configuration set for which you want to create a CloudWatch event destination. If the list is empty, you must first create a configuration set (p. 290).
4. On the Event Destinations tab, for Add Destination, choose Select a destination type, and then choose CloudWatch.
5. On the CloudWatch Destination dialog box, select Enabled.
6. For Name, type a name for the event destination.
7. For Event types, select the event types you want to publish to the event destination. The following event types are available:
   - Sends – The call to Amazon SES was successful and Amazon SES will attempt to deliver the email.
   - Rejects – Amazon SES accepted the email, determined that it contained a virus, and rejected it. Amazon SES didn’t attempt to deliver the email to the recipient’s mail server.
   - Bounces – The recipient’s mail server permanently rejected the email. This event corresponds to hard bounces. Soft bounces are only included when Amazon SES fails to deliver the email after retrying for a period of time.
   - Complaints – The email was successfully delivered to the recipient. The recipient marked the email as spam.
   - Deliveries – Amazon SES successfully delivered the email to the recipient’s mail server.
   - Opens – The recipient received the message and opened it in their email client.
   - Clicks – The recipient clicked one or more links in the email.
   - Rendering Failures – The email wasn’t sent because of a template rendering issue. This event type only occurs when you send email using the SendTemplatedEmail or SendBulkTemplatedEmail API operations. This event type can occur when template data is missing, or when there is a mismatch between template parameters and data.
   - Delivery Delays – The email couldn’t be delivered to the recipient because a temporary issue occurred. Delivery delays can occur, for example, when the recipient’s inbox is full, or when the receiving email server experiences a transient issue.
8. For Value Source, specify how Amazon SES will obtain the data that it passes to CloudWatch. The following value sources are available:
   - Message Tag – Amazon SES retrieves the dimension name and value from a tag that you specify by using the X-SES-MESSAGE-TAGS header or the Tags API parameter. For more information about using message tags, see the section called “Step 3: Specify your configuration set when sending” (p. 298).
     
     **Note**
     Message tags can include the numbers 0–9, the letters A–Z (both uppercase and lowercase), hyphens (-), and underscores (_).

     You can also use the Message Tag value source to create dimensions based on Amazon SES auto-tags. To use an auto-tag, type the complete name of the auto-tag as the Dimension Name. For example, to create a dimension based on the configuration set auto-tag, use ses:configuration-set for the Dimension Name, and the name of the configuration set for the Default Value. For a complete list of auto-tags, see How event publishing works (p. 288).
   - Email Header – Amazon SES retrieves the dimension name and value from a header in the email.
     
     **Note**
     You can’t use any of the following email headers as the Dimension Name: Received, To, From, DKIM-Signature, CC, message-id, or Return-Path.
   - Link Tag – Amazon SES retrieves the dimension name and value from a tag that you specified in a link. For more information about adding tags to links, see Can I tag links with unique identifiers? (p. 508).
9. For **Dimension Name**, type the name of the dimension that you want to pass to CloudWatch. For **Default Value**, type the value of the dimension.

   **Note**
   Dimension names and values can only contain the letters A through Z, the numbers 0 through 9, underscores (_), at signs (@), hyphens (-), and periods (.). Spaces, accented characters, non-Latin characters, and other special characters are not allowed.

10. If you want to add more dimensions, choose **Add Dimension**. Otherwise, choose **Save**.

**Choosing CloudWatch Dimensions**

When you choose names and values to use as CloudWatch dimensions, consider the following factors:

- **Price per metric** – You can view basic Amazon SES metrics in CloudWatch for free. However, when you collect metrics using event publishing, you create *custom metrics* in CloudWatch. Each unique combination of event type, dimension name, and dimension value creates a different custom metric in CloudWatch. When you use CloudWatch, you are charged for each custom metric you create. For this reason, you might want to avoid choosing dimensions that can take many different values. For example, unless you are very interested in tracking your email sending events by "From" domain, you might not want to define a dimension for the Amazon SES auto-tag `ses:from-domain` because it can take many different values. For more information, see CloudWatch Pricing.

- **Metric filtering** – If a metric has multiple dimensions, you cannot access the metric in CloudWatch based on each dimension separately. For that reason, think carefully before you add more than one dimension to a single CloudWatch event destination. For example, if you want metrics by campaign and by a combination of campaign and genre, you need to add two event destinations: one with only campaign as a dimension, and one with both campaign and genre as dimensions.

- **Dimension value source** – As an alternative to specifying your dimension values using Amazon SES-specific headers or a parameter to the API, you can also choose for Amazon SES to take the dimension values from your own MIME message headers. You might use this option if you are already using custom headers and you do not want to change your emails or your calls to the email sending API to collect metrics based on your header values. If you use your own MIME message headers for Amazon SES event publishing, the header names and values that you use for Amazon SES event publishing may only include the letters A through Z, the numbers 0 through 9, underscores (_), at signs (@), hyphens (-), and periods (.). If you specify a name or value that contains other characters, the email sending call will still succeed, but the event metrics will not be sent to Amazon CloudWatch.

For more information about CloudWatch concepts, see Amazon CloudWatch Concepts in the Amazon CloudWatch User Guide.

**Set up a Kinesis Data Firehose event destination for Amazon SES event publishing**

An Amazon Kinesis Data Firehose event destination represents an entity that publishes specific Amazon SES email sending events to Kinesis Data Firehose. Because a Kinesis Data Firehose event destination exists within a configuration set only, you first have to create a configuration set (p. 290). Next, you add the event destination to the configuration set.

This section includes a procedure for creating an event destination by using the Amazon SES console. You can also use the `UpdateConfigurationSetEventDestination` operation in the Amazon SES API V2 destination to create and update event destinations.

**To add a Kinesis Data Firehose event destination to a configuration set (console)**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, choose Configuration Sets.
3. Choose a configuration set from the configuration set list. If the list is empty, you must first create a configuration set (p. 290).
4. For Add Destination, choose Select a destination type, and then choose Kinesis Data Firehose.
5. For Name, type a name for the event destination.
6. For Event types, select at least one event type to publish to the event destination:
   - Sends – The call to Amazon SES was successful and Amazon SES will attempt to deliver the email.
   - Rejects – Amazon SES accepted the email, determined that it contained a virus, and rejected it. Amazon SES didn't attempt to deliver the email to the recipient's mail server.
   - Bounces – The recipient's mail server permanently rejected the email. This event corresponds to hard bounces. Soft bounces are only included when Amazon SES fails to deliver the email after retrying for a period of time.
   - Complaints – The email was successfully delivered to the recipient. The recipient marked the email as spam.
   - Deliveries – Amazon SES successfully delivered the email to the recipient's mail server.
   - Opens – The recipient received the message and opened it in their email client.
   - Clicks – The recipient clicked one or more links in the email.
   - Rendering Failures – The email wasn't sent because of a template rendering issue. This event type only occurs when you send email using the SendTemplatedEmail or SendBulkTemplatedEmail API operations. This event type can occur when template data is missing, or when there is a mismatch between template parameters and data.
   - Delivery Delays – The email couldn't be delivered to the recipient because a temporary issue occurred. Delivery delays can occur, for example, when the recipient's inbox is full, or when the receiving email server experiences a transient issue.
7. Select Enabled.
8. For Stream, choose an existing Kinesis Data Firehose delivery stream, or choose Create new stream to create a new one using the Kinesis Data Firehose console.
   For information about creating a stream using the Kinesis Data Firehose console, see Creating an Amazon Kinesis Firehose Delivery Stream in the Amazon Kinesis Data Firehose Developer Guide.
9. For IAM role, choose an IAM role for which Amazon SES has permission to publish to Kinesis Data Firehose on your behalf. You can choose an existing role, have Amazon SES create a role for you, or create your own role.
   If you choose an existing role or create your own role, you must manually modify the role's policies to give the role permission to access the Kinesis Data Firehose delivery stream, and to give Amazon SES permission to assume the role. For example policies, see Giving Amazon SES Permission to Publish to Your Kinesis Data Firehose Delivery Stream (p. 294).
10. Choose Save.
   For information about how to use the UpdateConfigurationSetEventDestination API to add a Kinesis Data Firehose event destination, see the Amazon Simple Email Service API Reference.

Giving Amazon SES Permission to Publish to Your Kinesis Data Firehose Delivery Stream

To enable Amazon SES to publish records to your Kinesis Data Firehose delivery stream, you must use an AWS Identity and Access Management (IAM) role and attach or modify the role's permissions policy and trust policy. The permissions policy enables the role to publish records to your Kinesis Data Firehose delivery stream, and the trust policy enables Amazon SES to assume the role.

This section provides examples of both policies. For information about attaching policies to IAM roles, see Modifying a Role in the IAM User Guide.
Permissions Policy

The following permissions policy enables the role to publish data records to your Kinesis Data Firehose delivery stream.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Sid": "",
        "Effect": "Allow",
        "Action": ["firehose:PutRecordBatch"],
    }
    ]
}
```

Make the following changes to the preceding policy example:

- If you're located in a special partition such as China or AWS GovCloud (US), replace `aws` with the corresponding AWS partition, see Amazon Resource Names (ARNs) in the AWS General Reference.
- Replace `us-west-2` with the AWS Region where you created the Kinesis Data Firehose delivery stream.
- Replace `111122223333` with your AWS account ID.
- Replace `delivery-stream-name` with the name of the Kinesis Data Firehose delivery stream.

Trust Policy

The following trust policy enables Amazon SES to assume the role.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Sid": "",
        "Effect": "Allow",
        "Principal": {"Service": "ses.amazonaws.com"},
        "Action": "sts:AssumeRole",
        "Condition": {
            "StringEquals": {
                "AWS:SourceAccount": "111122223333",
            }
        }
    }
    ]
}
```

Make the following changes to the preceding policy example:

- If you're located in a special partition such as China or AWS GovCloud (US), replace `aws` with the corresponding AWS partition, see Amazon Resource Names (ARNs) in the AWS General Reference.
• Replace `us-west-2` with the AWS Region where you created the Kinesis Data Firehose delivery stream.
• Replace `111122223333` with your AWS account ID.
• Replace `configuration-set-name` with the name of your configuration set associated with the Kinesis Data Firehose delivery stream.

Set up an Amazon SNS event destination for event publishing

A SNS event destination notifies you about specific email sending events using Amazon SNS. Because an Amazon SNS event destination only exists within a configuration set, you have to create a configuration set (p. 290) before you add the event destination to the configuration set.

This section includes a procedure for creating an event destination by using the Amazon SES console. You can also use the `UpdateConfigurationSetEventDestination` operation in the Amazon SES API V2 destination to create and update event destinations.

**Note**

It's also possible to receive notifications through Amazon SNS at the account level. This means that you can receive Amazon SNS notifications every time a sending event occurs across your entire Amazon SES account. By using event publishing rather than account-level notifications, you can configure Amazon SES to only send notifications about specific event types, or only for emails sent using a particular configuration set. For more information about setting up account-level Amazon SNS notifications, see Monitoring Amazon SES email sending using notifications (p. 265).

There are additional charges for sending messages to the endpoints that are subscribed to your Amazon SNS topics. For more information, see Amazon SNS Pricing.

To add an Amazon SNS event destination to a configuration set

1. If you have not already done so, create an Amazon SNS topic and subscribe to it. For more information, see Create a Topic in the Amazon Simple Notification Service Developer Guide.

   **Important**
   
   When you create your topic using Amazon SNS, for Type, only choose Standard. (SES does not support FIFO type topics.)

2. Open the Amazon SES console at `https://console.aws.amazon.com/ses/`.
3. In the navigation pane, choose Configuration Sets.
4. Choose a configuration set from the configuration set list. If the list is empty, you must first create a configuration set (p. 290).
5. For Add Destination, choose Select a destination type, and then choose SNS.
6. Ensure the Enabled field's check box is pre-selected.
7. For Name, type a name for the event destination.
8. For Event types, select at least one event type to publish to the event destination:

   • Sends – The call to Amazon SES was successful and Amazon SES will attempt to deliver the email.
   • Rejects – Amazon SES accepted the email, determined that it contained a virus, and rejected it. Amazon SES didn't attempt to deliver the email to the recipient's mail server.
   • Bounces – The recipient's mail server permanently rejected the email. This event corresponds to hard bounces. Soft bounces are only included when Amazon SES fails to deliver the email after retrying for a period of time.
   • Complaints – The email was successfully delivered to the recipient. The recipient marked the email as spam.
   • Deliveries – Amazon SES successfully delivered the email to the recipient's mail server.
   • Opens – The recipient received the message and opened it in their email client.
• **Clicks** – The recipient clicked one or more links in the email.

• **Rendering Failures** – The email wasn't sent because of a template rendering issue. This event type only occurs when you send email using the `SendTemplatedEmail` or `SendBulkTemplatedEmail` API operations. This event type can occur when template data is missing, or when there is a mismatch between template parameters and data.

• **Delivery Delays** – The email couldn't be delivered to the recipient because a temporary issue occurred. Delivery delays can occur, for example, when the recipient's inbox is full, or when the receiving email server experiences a transient issue.

9. For **Topic**, choose an existing Amazon SNS topic, or choose **Create new topic** to create a new one.

For information about creating a topic, see [Create a Topic](http://aws.amazon.com/documentation/sns/#Create_a_Topic) in the *Amazon Simple Notification Service Developer Guide*.

**Important**

When you create your topic using Amazon SNS, for **Type**, only choose **Standard**. (SES does not support FIFO type topics.)

Whether you create a new SNS topic or select an existing one, you need to give access to SES to publish notifications to the topic.

To give Amazon SES permission to publish notifications to the topic, on the **Edit topic** screen in the SNS console, expand **Access policy** and in the **JSON editor**, add the following permission policy:

```json
{
  "Version": "2012-10-17",
  "Id": "notification-policy",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ses.amazonaws.com"
      },
      "Action": "sns:Publish",
      "Condition": {
        "StringEquals": {
          "AWS:SourceAccount": "111122223333",
        }
      }
    }
  ]
}
```

Make the following changes to the preceding policy example:

- If you're located in a special partition such as China or AWS GovCloud (US), replace `aws` with the corresponding AWS partition, see [Amazon Resource Names (ARNs)](http://aws.amazon.com/documentation/general/latest/arns/index.html) in the *AWS General Reference*.
- Replace `us-west-2` with the AWS Region where you created the SNS topic.
- Replace `111122223333` with your AWS account ID.
- Replace `topic_name` with the name of your SNS topic.
- Replace `configuration-set-name` with the name of your configuration set associated with the SNS event destination.

10. Choose **Save**.

11. To use a configuration set when sending an email, see [Specifying a configuration set when you send email](http://aws.amazon.com/documentation/ses/#Specifying_a_configuration_set_when_you_send_email) (p. 257).
Step 3: Specify your configuration set when you send email

After you create a configuration set (p. 290) and add an event destination (p. 291), the last step to event publishing is to send your emails.

To publish events associated with an email, you must provide the name of the configuration set to associate with the email. Optionally, you can provide message tags to categorize the email.

You provide this information to Amazon SES as either parameters to the email sending API, Amazon SES-specific email headers, or custom headers in your MIME message. The method you choose depends on which email sending interface you use, as shown in the following table.

<table>
<thead>
<tr>
<th>Email Sending Interface</th>
<th>Ways to Publish Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>SendEmail</td>
<td>API parameters</td>
</tr>
<tr>
<td>SendRawEmail</td>
<td>API parameters, Amazon SES-specific email headers, or custom MIME headers</td>
</tr>
<tr>
<td></td>
<td><strong>Important</strong></td>
</tr>
<tr>
<td></td>
<td>If you specify message tags using both headers and API parameters, Amazon SES uses only the message tags provided by the API parameters. Amazon SES does not join message tags specified by API parameters and headers.</td>
</tr>
<tr>
<td>SMTP interface</td>
<td>Amazon SES-specific email headers</td>
</tr>
</tbody>
</table>

The following sections describe how to specify the configuration set and message tags using headers and using API parameters.

- Using Amazon SES API Parameters (p. 298)
- Using Amazon SES-Specific Email Headers (p. 299)
- Using Custom Email Headers (p. 299)

Additionally, this guide contains several code examples that demonstrate how to send email programmatically using Amazon SES. Each of these code examples includes a method of passing a configuration set when sending an email. For more information, see Amazon SES code examples (p. 416).

Note
You can optionally include message tags in the headers of your emails. Message tags can include the numbers 0–9, the letters A–Z (both uppercase and lowercase), hyphens (-), and underscores (_).

Using Amazon SES API Parameters

To use SendEmail or SendRawEmail with event publishing, you specify the configuration set and the message tags by passing data structures called ConfigurationSet and MessageTag to the API call.

For more information about using the Amazon SES API, see the Amazon Simple Email Service API Reference.
Using Amazon SES-Specific Email Headers

When you use `SendRawEmail` or the SMTP interface, you can specify the configuration set and the message tags by adding Amazon SES-specific headers to the email. Amazon SES removes the headers before sending the email. The following table shows the names of the headers to use.

<table>
<thead>
<tr>
<th>Event Publishing Information</th>
<th>Header</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration set</td>
<td>X-SES-CONFIGURATION-SET</td>
</tr>
<tr>
<td>Message tags</td>
<td>X-SES-MESSAGE-TAGS</td>
</tr>
</tbody>
</table>

The following example shows how the headers might look in a raw email that you submit to Amazon SES.

```plaintext
X-SES-MESSAGE-TAGS: tagName1=tagValue1, tagName2=tagValue2
X-SES-CONFIGURATION-SET: myConfigurationSet
From: sender@example.com
To: recipient@example.com
Subject: Subject
Content-Type: multipart/alternative;
boundary="----=_boundary"

----=_boundary
Content-Type: text/plain; charset=UTF-8
Content-Transfer-Encoding: 7bit

body

----=_boundary
Content-Type: text/html; charset=UTF-8
Content-Transfer-Encoding: 7bit

body

----=_boundary--
```

Using Custom Email Headers

Although you must specify the configuration set name using the Amazon SES-specific header `X-SES-CONFIGURATION-SET`, you can specify the message tags by using your own MIME headers.

**Note**
Header names and values that you use for Amazon SES event publishing must be in ASCII. If you specify a non-ASCII header name or value for Amazon SES event publishing, the email sending call will still succeed, but the event metrics will not be emitted to Amazon CloudWatch.

Working with Amazon SES event data

After you set up event publishing (p. 290) and specify a configuration set for sending emails, you can retrieve your email sending events from the event destination that you specified when you set up the configuration set associated with the email.

This section describes how to retrieve your email sending events from Amazon CloudWatch and Amazon Kinesis Data Firehose, and how to interpret event data provided by Amazon SNS.

- Retrieving Amazon SES event data from CloudWatch (p. 300)
- Retrieving Amazon SES event data from Kinesis Data Firehose (p. 301)
- Interpreting Amazon SES event data from Amazon SNS (p. 321)
Retrieving Amazon SES event data from CloudWatch

Amazon SES can publish metrics for your email sending events to Amazon CloudWatch. When you publish event data to CloudWatch, it provides these metrics as an ordered set of time-series data. You can use these metrics to monitor the performance of your email sending. For example, you can monitor the complaint metric and set a CloudWatch alarm to trigger when the metric exceeds a certain value.

There are two levels of granularity at which Amazon SES can publish these events to CloudWatch:

- **Across your AWS account** – These coarse metrics, which correspond to the metrics you monitor using the Amazon SES console and the `GetSendStatistics` API, are totals across your entire AWS account. Amazon SES publishes these metrics to CloudWatch automatically.
- **Fine-grained** – These metrics are categorized by email characteristics that you define using message tags. To publish these metrics to CloudWatch, you have to set up event publishing (p. 290) with a CloudWatch event destination and specify a configuration set (p. 298) when you send an email. You can also specify message tags or use auto-tags (p. 288) that Amazon SES automatically provides.

This section describes the available metrics and how to view the metrics in CloudWatch.

### Available Metrics

You can publish following Amazon SES email sending metrics to CloudWatch:

- **Sends** – The call to Amazon SES was successful and Amazon SES will attempt to deliver the email.
- **Rejects** – Amazon SES accepted the email, determined that it contained a virus, and rejected it. Amazon SES didn't attempt to deliver the email to the recipient's mail server.
- **Bounces** – The recipient's mail server permanently rejected the email. This event corresponds to hard bounces. Soft bounces are only included when Amazon SES fails to deliver the email after retrying for a period of time.
- **Complaints** – The email was successfully delivered to the recipient. The recipient marked the email as spam.
- **Deliveries** – Amazon SES successfully delivered the email to the recipient's mail server.
- **Opens** – The recipient received the message and opened it in their email client.
- **Clicks** – The recipient clicked one or more links in the email.
- **Rendering Failures** – The email wasn't sent because of a template rendering issue. This event type only occurs when you send email using the `SendTemplatedEmail` or `SendBulkTemplatedEmail` API operations. This event type can occur when template data is missing, or when there is a mismatch between template parameters and data.
- **Delivery Delays** – The email couldn't be delivered to the recipient because a temporary issue occurred. Delivery delays can occur, for example, when the recipient's inbox is full, or when the receiving email server experiences a transient issue.

### Available Dimensions

CloudWatch uses the dimension names that you specify when you add a CloudWatch event destination to a configuration set in Amazon SES. For more information, see Set up a CloudWatch event destination for event publishing (p. 291).

### Viewing Amazon SES Metrics in the CloudWatch Console

The following procedure describes how to view your Amazon SES event publishing metrics using the CloudWatch console.
To view metrics using the CloudWatch console

1. Sign in to the AWS Management Console and open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. If necessary, change the region. From the navigation bar, select the region where your AWS resources reside. For more information, see Regions and Endpoints.
3. In the navigation pane, choose Metrics.
4. In the All metrics pane, expand AWS Namespaces, and then choose SES.
5. To view metrics across your entire AWS account, which Amazon SES publishes automatically, choose Account Sending Metrics. To view fine-grained event publishing metrics (p. 288), choose the combination of dimensions that you specified when you set up your CloudWatch event destination (p. 291).
6. Choose the metric you want to view.
   The graph will display the metric over time.

To view metrics using the AWS CLI

• At a command prompt, use the following command:

   ```bash
   aws cloudwatch list-metrics --namespace "AWS/SES"
   ```

Retrieving Amazon SES event data from Kinesis Data Firehose

Amazon SES publishes email sending events to Kinesis Data Firehose as JSON records. Kinesis Data Firehose then publishes the records to the AWS service destination that you chose when you set up the delivery stream in Kinesis Data Firehose. For information about setting up Kinesis Data Firehose delivery streams, see Creating an Kinesis Data Firehouse Delivery Stream in the Amazon Kinesis Data Firehose Developer Guide.

For examples of how you can use Kinesis Data Firehose to publish your email sending events to Amazon Redshift and Amazon OpenSearch Service, see Tutorials (p. 341).

Topics in this section:
• Contents of event data that Amazon SES publishes to Kinesis Data Firehose (p. 301)
• Examples of event data that Amazon SES publishes to Kinesis Data Firehose (p. 310)

Contents of event data that Amazon SES publishes to Kinesis Data Firehose

Amazon SES publishes email sending event records to Amazon Kinesis Data Firehose in JSON format. When publishing events to Kinesis Data Firehose, Amazon SES follows each JSON record with a newline character.

The top-level JSON object contains an eventType string, a mail object, and either a Bounce, Complaint, Delivery, Send, Reject, Open, Click, Rendering Failure, or DeliveryDelay object, depending on the type of event.

You can find example records for all of these notification types in Examples of event data that Amazon SES publishes to Kinesis Data Firehose (p. 310).

Topics in this section
• Top-level JSON object (p. 302)
Top-level JSON object

The top-level JSON object in an email sending event record contains the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eventType</td>
<td>A string that describes the type of event. Possible values: Delivery, Send, Reject, Open, Click, Bounce, Complaint, Rendering Failure, or DeliveryDelay.</td>
</tr>
<tr>
<td>mail</td>
<td>A JSON object that contains information about the email that produced the event.</td>
</tr>
<tr>
<td>bounce</td>
<td>This field is only present if eventType is Bounce. It contains information about the bounce.</td>
</tr>
<tr>
<td>complaint</td>
<td>This field is only present if eventType is Complaint. It contains information about the complaint.</td>
</tr>
<tr>
<td>delivery</td>
<td>This field is only present if eventType is Delivery. It contains information about the delivery.</td>
</tr>
<tr>
<td>send</td>
<td>This field is only present if eventType is Send.</td>
</tr>
<tr>
<td>reject</td>
<td>This field is only present if eventType is Reject. It contains information about the rejection.</td>
</tr>
<tr>
<td>open</td>
<td>This field is only present if eventType is Open. It contains information about the open event.</td>
</tr>
<tr>
<td>click</td>
<td>This field is only present if eventType is Click. It contains information about the click event.</td>
</tr>
<tr>
<td>failure</td>
<td>This field is only present if eventType is Rendering Failure. It contains information about the rendering failure event.</td>
</tr>
<tr>
<td>deliveryDelay</td>
<td>This field is only present if eventType is DeliveryDelay. It contains information about the delayed delivery of an email.</td>
</tr>
</tbody>
</table>
Mail object

Each email sending event record contains information about the original email in the mail object. The JSON object that contains information about a mail object has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timestamp</td>
<td>The date and time, in ISO8601 format (YYYY-MM-DDThh:mm:ss.sZ), when the message was sent.</td>
</tr>
</tbody>
</table>
| messageId        | A unique ID that Amazon SES assigned to the message. Amazon SES returned this value to you when you sent the message.  
Note  
This message ID was assigned by Amazon SES. You can find the message ID of the original email in the headers and commonHeaders fields of the mail object. |
| source           | The email address that the message was sent from (the envelope MAIL FROM address). |
| sourceArn        | The Amazon Resource Name (ARN) of the identity that was used to send the email. In the case of sending authorization, the sourceArn is the ARN of the identity that the identity owner authorized the delegate sender to use to send the email. For more information about sending authorization, see Using sending authorization (p. 149). |
| sendingAccountId | The AWS account ID of the account that was used to send the email. In the case of sending authorization, the sendingAccountId is the delegate sender's account ID. |
| destination      | A list of email addresses that were recipients of the original mail. |
| headersTruncated | A string that specifies whether the headers are truncated in the notification, which occurs if the headers are larger than 10 KB. Possible values are true and false. |
| headers          | A list of the email's original headers. Each header in the list has a name field and a value field.  
Note  
Any message ID within the headers field is from the original message that you passed to Amazon SES. The message ID that Amazon SES subsequently assigned to the message is in the messageId field of the mail object. |
| commonHeaders    | A list of the email's original, commonly used headers. Each header in the list has a name field and a value field. |
Field Name | Description
--- | ---
**Note** | Any message ID within the `commonHeaders` field is from the original message that you passed to Amazon SES. The message ID that Amazon SES subsequently assigned to the message is in the `messageId` field of the `mail` object.

**Bounce object**

The JSON object that contains information about a Bounce event will always have the following fields.

Field Name | Description
--- | ---
bounceType | The type of bounce, as determined by Amazon SES.
bounceSubType | The subtype of the bounce, as determined by Amazon SES.
bouncedRecipients | A list that contains information about the recipients of the original mail that bounced.
timestamp | The date and time, in ISO8601 format (`YYYY-MM-DDThh:mm:ss.sZ`), when the ISP sent the bounce notification.
feedbackId | A unique ID for the bounce.
reportingMTA | The value of the `Reporting-MTA` field from the DSN. This is the value of the Message Transfer Authority (MTA) that attempted to perform the delivery, relay, or gateway operation described in the DSN.

**Note**
This field only appears if a delivery status notification (DSN) was attached to the bounce.

**Bounced recipients**

A bounce event may pertain to a single recipient or to multiple recipients. The `bouncedRecipients` field holds a list of objects—one object per recipient to whom the bounce event pertains—and will always contain the following field.

Field Name | Description
--- | ---
emailAddress | The email address of the recipient. If a DSN is available, this is the value of the `Final-Recipient` field from the DSN.

Optionally, if a DSN is attached to the bounce, the following fields may also be present.
Field Name | Description
---|---
action | The value of the `Action` field from the DSN. This indicates the action performed by the reporting MTA as a result of its attempt to deliver the message to this recipient.
status | The value of the `Status` field from the DSN. This is the per-recipient transport-independent status code that indicates the delivery status of the message.
diagnosticCode | The status code issued by the reporting MTA. This is the value of the `Diagnostic-Code` field from the DSN. This field may be absent in the DSN (and therefore also absent in the JSON).

Bounce types

Each bounce event will be of one of the types shown in the following table.

The event publishing system only publishes hard bounces and soft bounces that will no longer be retried by Amazon SES. When you receive bounces marked `Permanent`, you should remove the corresponding email addresses from your mailing list; you will not be able to send to them in the future. `Transient` bounces are sent to you when a message has soft bounced several times, and Amazon SES has stopped trying to re-deliver it. You may be able to successfully resend to an address that initially resulted in a `Transient` bounce in the future.

<table>
<thead>
<tr>
<th>bounceType</th>
<th>bounceSubType</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Undetermined</td>
<td>Undetermined</td>
<td>Amazon SES was unable to determine a specific bounce reason.</td>
</tr>
<tr>
<td>Permanent</td>
<td>General</td>
<td>Amazon SES received a general hard bounce. If you receive this type of bounce, you should remove the recipient’s email address from your mailing list.</td>
</tr>
<tr>
<td>Permanent</td>
<td>NoEmail</td>
<td>Amazon SES received a permanent hard bounce because the target email address does not exist. If you receive this type of bounce, you should remove the recipient’s email address from your mailing list.</td>
</tr>
<tr>
<td>Permanent</td>
<td>Suppressed</td>
<td>Amazon SES has suppressed sending to this address because it has a recent history of bouncing as an invalid address. For information about how to remove an address from the suppression list, see Using the Amazon SES global suppression list (p. 192).</td>
</tr>
<tr>
<td>Permanent</td>
<td>OnAccountSuppressionList</td>
<td>Amazon SES has suppressed sending to this address because it is on the account-level suppression list (p. 184).</td>
</tr>
</tbody>
</table>
### Working with event data

<table>
<thead>
<tr>
<th>bounceType</th>
<th>bounceSubType</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transient</td>
<td>General</td>
<td>Amazon SES received a general bounce. You may be able to successfully send to this recipient in the future.</td>
</tr>
<tr>
<td>Transient</td>
<td>MailboxFull</td>
<td>Amazon SES received a mailbox full bounce. You may be able to successfully send to this recipient in the future.</td>
</tr>
<tr>
<td>Transient</td>
<td>MessageTooLarge</td>
<td>Amazon SES received a message too large bounce. You may be able to successfully send to this recipient if you reduce the size of the message.</td>
</tr>
<tr>
<td>Transient</td>
<td>ContentRejected</td>
<td>Amazon SES received a content rejected bounce. You may be able to successfully send to this recipient if you change the content of the message.</td>
</tr>
<tr>
<td>Transient</td>
<td>AttachmentRejected</td>
<td>Amazon SES received an attachment rejected bounce. You may be able to successfully send to this recipient if you remove or change the attachment.</td>
</tr>
</tbody>
</table>

#### Complaint object

The JSON object that contains information about a complaint event has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>complainedRecipients</td>
<td>A list that contains information about recipients that may have submitted the complaint.</td>
</tr>
<tr>
<td>timestamp</td>
<td>The date and time, in ISO8601 format (YYYY-MM-DDThh:mm:ss.sZ), when the ISP sent the complaint notification.</td>
</tr>
<tr>
<td>feedbackId</td>
<td>A unique ID for the complaint.</td>
</tr>
<tr>
<td>complaintSubType</td>
<td>The subtype of the complaint, as determined by Amazon SES.</td>
</tr>
</tbody>
</table>

Further, if a feedback report is attached to the complaint, the following fields may be present.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userAgent</td>
<td>The value of the User-Agent field from the feedback report. This indicates the name and version of the system that generated the report.</td>
</tr>
<tr>
<td>complaintFeedbackType</td>
<td>The value of the Feedback-Type field from the feedback report received from the ISP. This contains the type of feedback.</td>
</tr>
<tr>
<td>arrivalDate</td>
<td>The value of the Arrival-Date or Received-Date field from the feedback report in ISO8601</td>
</tr>
</tbody>
</table>
## Field Name | Description
--- | ---
| format (YYYY-MM-DDThh:mm:ss.sZ). This field may be absent in the report (and therefore also absent in the JSON). |  

### Complained recipients

The `complainedRecipients` field contains a list of recipients that may have submitted the complaint.

**Important**
Since most ISPs redact the email address of the recipient who submitted the complaint from their complaint notification, this list contains information about recipients who might have sent the complaint, based on the recipients of the original message and the ISP from which we received the complaint. Amazon SES performs a lookup against the original message to determine this recipient list.

JSON objects in this list contain the following field.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>emailAddress</td>
<td>The email address of the recipient.</td>
</tr>
</tbody>
</table>

### Complaint types

You may see the following complaint types in the `complaintFeedbackType` field as assigned by the reporting ISP, according to the Internet Assigned Numbers Authority website:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>abuse</td>
<td>Indicates unsolicited email or some other kind of email abuse.</td>
</tr>
<tr>
<td>auth-failure</td>
<td>Email authentication failure report.</td>
</tr>
<tr>
<td>fraud</td>
<td>Indicates some kind of fraud or phishing activity.</td>
</tr>
<tr>
<td>not-spam</td>
<td>Indicates that the entity providing the report does not consider the message to be spam. This may be used to correct a message that was incorrectly tagged or categorized as spam.</td>
</tr>
<tr>
<td>other</td>
<td>Indicates any other feedback that does not fit into other registered types.</td>
</tr>
<tr>
<td>virus</td>
<td>Reports that a virus is found in the originating message.</td>
</tr>
</tbody>
</table>

### Delivery object

The JSON object that contains information about a Delivery event will always have the following fields.
### Field Name | Description
--- | ---
timestamp | The date and time when Amazon SES delivered the email to the recipient's mail server, in ISO8601 format (YYYY-MM-DDThh:mm:ss.sZ).
processingTimeMillis | The time in milliseconds between when Amazon SES accepted the request from the sender to when Amazon SES passed the message to the recipient's mail server.
recipients | A list of intended recipients that the delivery event applies to.
smtpResponse | The SMTP response message of the remote ISP that accepted the email from Amazon SES. This message will vary by email, by receiving mail server, and by receiving ISP.
reportingMTA | The host name of the Amazon SES mail server that sent the mail.

### Send object
The JSON object that contains information about a send event is always empty.

### Reject object
The JSON object that contains information about a reject event will always have the following fields.

| Field Name | Description |
--- | ---
reason | The reason the email was rejected. The only possible value is Bad content, which means that Amazon SES detected that the email contained a virus. When a message is rejected, Amazon SES stops processing it, and doesn't attempt to deliver it to the recipient's mail server. |

### Open object
The JSON object that contains information about a open event will always contain the following fields.

| Field Name | Description |
--- | ---
ipAddress | The recipient's IP address. |
timestamp | The date and time when the open event occurred in ISO8601 format (YYYY-MM-DDThh:mm:ss.sZ). |
userAgent | The user agent of the device or email client that the recipient used to open the email. |
Click object

The JSON object that contains information about a click event will always contain the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ipAddress</td>
<td>The recipient's IP address.</td>
</tr>
<tr>
<td>timestamp</td>
<td>The date and time when the click event occurred in ISO8601 format (YYYY-MM-DDThh:mm:ss.sZ).</td>
</tr>
<tr>
<td>userAgent</td>
<td>The user agent of the client that the recipient used to click a link in the email.</td>
</tr>
<tr>
<td>link</td>
<td>The URL of the link that the recipient clicked.</td>
</tr>
<tr>
<td>linkTags</td>
<td>A list of tags that were added to the link using the ses:tags attribute. For more information about adding tags to links in your emails, see Q5. Can I tag links with unique identifiers? (p. 508) in the Amazon SES email sending metrics FAQs (p. 506).</td>
</tr>
</tbody>
</table>

Rendering Failure object

The JSON object that contains information about a Rendering Failure event has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>templateName</td>
<td>The name of the template used to send the email.</td>
</tr>
<tr>
<td>errorMessage</td>
<td>A message that provides more information about the rendering failure.</td>
</tr>
</tbody>
</table>

DeliveryDelay object

The JSON object that contains information about a DeliveryDelay event has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>delayType</td>
<td>The type of delay. Possible values are:</td>
</tr>
<tr>
<td></td>
<td>• InternalFailure – An internal Amazon SES issue caused the message to be delayed.</td>
</tr>
<tr>
<td></td>
<td>• General – A generic failure occurred during the SMTP conversation.</td>
</tr>
<tr>
<td></td>
<td>• MailboxFull – The recipient's mailbox is full and is unable to receive additional messages.</td>
</tr>
<tr>
<td></td>
<td>• SpamDetected – The recipient's mail server has detected a large amount of unsolicited email from your account.</td>
</tr>
<tr>
<td></td>
<td>• RecipientServerError – A temporary issue with the recipient's email server is preventing the delivery of the message.</td>
</tr>
</tbody>
</table>
### Field Name | Description
---|---
- IPFailure – The IP address that's sending the message is being blocked or throttled by the recipient's email provider.
- TransientCommunicationGeneral – There was a temporary communication failure during the SMTP conversation with the recipient's email provider.
- BYOIPHostNameLookupUnavailable – Amazon SES was unable to look up the DNS hostname for your IP addresses. This type of delay only occurs when you use Bring Your Own IP (p. 180).
- Undetermined – Amazon SES wasn't able to determine the reason for the delivery delay.

#### delayedRecipients
An object that contains information about the recipient of the email.

#### expirationTime
The date and time when Amazon SES will stop trying to deliver the message. This value is shown in ISO 8601 format.

#### reportingMTA
The IP address of the Message Transfer Agent (MTA) that reported the delay.

#### timestamp
The date and time when the delay occurred, shown in ISO 8601 format.

### Delayed recipients
The `delayedRecipients` object contains the following values.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>emailAddress</td>
<td>The email address that resulted in the delivery of the message being delayed.</td>
</tr>
<tr>
<td>status</td>
<td>The SMTP status code associated with the delivery delay.</td>
</tr>
<tr>
<td>diagnosticCode</td>
<td>The diagnostic code provided by the receiving Message Transfer Agent (MTA).</td>
</tr>
</tbody>
</table>

### Examples of event data that Amazon SES publishes to Kinesis Data Firehose

This section provides examples of the types of email sending event record that Amazon SES publishes to Kinesis Data Firehose.

#### Topics in this section:
- Bounce record (p. 311)
- Complaint record (p. 312)
- Delivery record (p. 313)
- Send record (p. 315)
- Reject record (p. 316)
The following is an example of a Bounce event record that Amazon SES publishes to Kinesis Data Firehose.

```json
{
    "eventType":"Bounce",
    "bounce":{
        "bounceType":"Permanent",
        "bounceSubType":"General",
        "bouncedRecipients": [
            {
                "emailAddress":"recipient@example.com",
                "action":"failed",
                "status":"5.1.1",
                "diagnosticCode":"smtp; 550 5.1.1 user unknown"
            }
        ],
        "timestamp":"2017-08-05T00:41:02.669Z",
        "feedbackId":"01000157c44f053b-61b59c11-9236-11e6-8f96-7be8aexample-000000",
        "reportingMTA":"dsn; mta.example.com"
    },
    "mail":{
        "timestamp":"2017-08-05T00:40:02.012Z",
        "source":"Sender Name <sender@example.com>",
        "sourceArn":"arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
        "sendingAccountId":"123456789012",
        "messageId":"EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
        "destination": [
            "recipient@example.com"
        ],
        "headersTruncated":false,
        "headers": [
            {
                "name":"From",
                "value":"Sender Name <sender@example.com>"
            },
            {
                "name":"To",
                "value":"recipient@example.com"
            },
            {
                "name":"Subject",
                "value":"Message sent from Amazon SES"
            },
            {
                "name":"MIME-Version",
                "value":"1.0"
            },
            {
                "name":"Content-Type",
                "value":"multipart/alternative; boundary="----=_Part_7307378_1629847660.1516840721503"
            }
        ],
        "commonHeaders":{
            "from":
        }
    }
}
```
Complaint record

The following is an example of a Complaint event record that Amazon SES publishes to Kinesis Data Firehose.

```json
{
    "eventType":"Complaint",
    "complaint": {
        "complainedRecipients": [
            {
                "emailAddress": "recipient@example.com"
            }
        ],
        "timestamp": "2017-08-05T00:41:02.669Z",
        "feedbackId": "01000157c44f053b-61b59c11-9236-11e6-8f96-7be8aexample-000000",
        "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/60.0.3112.90 Safari/537.36",
        "complaintFeedbackType": "abuse",
        "arrivalDate": "2017-08-05T00:41:02.669Z"
    },
    "mail": {
        "timestamp": "2017-08-05T00:40:01.123Z",
        "source": "Sender Name <sender@example.com>",
        "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
        "sendingAccountId": "123456789012",
        "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
        "destination": [
            "recipient@example.com"
        ],
        "headersTruncated": false,
        "headers": [
            {
                "name": "From",
                "value": "Sender Name <sender@example.com>"
            },
            {
                "name": "To",
                "value": "recipient@example.com"
            }
        ]
    }
}
Working with event data

The following is an example of a Delivery event record that Amazon SES publishes to Kinesis Data Firehose.

```json
{
  "eventType": "Delivery",
  "mail": {
    "timestamp": "2016-10-19T23:20:52.240Z",
    "source": "sender@example.com",
    "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId": "123456789012",
    "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination": ["recipient@example.com"],
    "headersTruncated": false,
    "headers": [
      {
        "name": "From",
        "value": "sender@example.com"
      },
      {
        "name": "To",
        "value": "recipient@example.com"
      }
    ]
  }
}```
"value": "recipient@example.com",
},
  "name": "Subject",
  "value": "Message sent from Amazon SES"
},
  "name": "MIME-Version",
  "value": "1.0"
},
  "name": "Content-Type",
  "value": "text/html; charset=UTF-8"
},
  "name": "Content-Transfer-Encoding",
  "value": "7bit"
],
"commonHeaders": {
  "from": [
    "sender@example.com"
  ],
  "to": [
    "recipient@example.com"
  ],
  "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
  "subject": "Message sent from Amazon SES"
},
"tags": {
  "ses:configuration-set": [
    "ConfigSet"
  ],
  "ses:source-ip": [
    "192.0.2.0"
  ],
  "ses:from-domain": [
    "example.com"
  ],
  "ses:caller-identity": [
    "ses_user"
  ],
  "ses:outgoing-ip": [
    "192.0.2.0"
  ],
  "myCustomTag1": [
    "myCustomTagValue1"
  ],
  "myCustomTag2": [
    "myCustomTagValue2"
  ]
},
"delivery": {
  "timestamp": "2016-10-19T23:21:04.133Z",
  "processingTimeMillis": 11893,
  "recipients": [
    "recipient@example.com"
  ],
  "smtpResponse": "250 2.6.0 Message received",
  "reportingMTA": "mta.example.com"
}
Send record

The following is an example of a Send event record that Amazon SES publishes to Kinesis Data Firehose.

```json
{
  "eventType": "Send",
  "mail": {
    "timestamp": "2016-10-14T05:02:16.645Z",
    "source": "sender@example.com",
    "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId": "123456789012",
    "messageId": "EXAMPLE7c191be45-e9adb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination": [
      "recipient@example.com"
    ],
    "headersTruncated": false,
    "headers": [
      {
        "name": "From",
        "value": "sender@example.com"
      },
      {
        "name": "To",
        "value": "recipient@example.com"
      },
      {
        "name": "Subject",
        "value": "Message sent from Amazon SES"
      },
      {
        "name": "MIME-Version",
        "value": "1.0"
      },
      {
        "name": "Content-Type",
        "value": "multipart/mixed; boundary="----=_Part_0_716996660.1476421336341"
      },
      {
        "name": "X-SES-MESSAGE-TAGS",
        "value": "myCustomTag1=myCustomTagValue1, myCustomTag2=myCustomTagValue2"
      }
    ],
    "commonHeaders": {
      "from": ["sender@example.com"],
      "to": ["recipient@example.com"],
      "messageId": "EXAMPLE7c191be45-e9adb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "subject": "Message sent from Amazon SES"
    },
    "tags": {
      "ses:configuration-set": [
        "ConfigSet"
      ],
      "ses:source-ip": ["192.0.2.0"],
      "ses:from-domain": [
        "example.com"
      ],
      "ses:caller-identity": [
        "ses_user"
      ]
    }
}``
Reject record

The following is an example of a Reject event record that Amazon SES publishes to Kinesis Data Firehose.

```json
{
    "eventType": "Reject",
    "mail": {
        "timestamp": "2016-10-14T17:38:15.211Z",
        "source": "sender@example.com",
        "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
        "sendingAccountId": "123456789012",
        "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
        "destination": [
            "sender@example.com"
        ],
        "headersTruncated": false,
        "headers": [
            {
                "name": "From",
                "value": "sender@example.com"
            },
            {
                "name": "To",
                "value": "recipient@example.com"
            },
            {
                "name": "Subject",
                "value": "Message sent from Amazon SES"
            },
            {
                "name": "MIME-Version",
                "value": "1.0"
            },
            {
                "name": "Content-Type",
                "value": "multipart/mixed; boundary="qMm9M+Fa2AknHoGS"
            },
            {
                "name": "X-SES-MESSAGE-TAGS",
                "value": "myCustomTag1=myCustomTagValue1, myCustomTag2=myCustomTagValue2"
            }
        ],
        "commonHeaders": {
            "from": ["sender@example.com"],
            "to": ["recipient@example.com"],
            "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
            "subject": "Message sent from Amazon SES"
        }
    }
}
```
"tags": {
  "ses:configuration-set": [
    "ConfigSet"
  ],
  "ses:source-ip": [
    "192.0.2.0"
  ],
  "ses:from-domain": [
    "example.com"
  ],
  "ses:caller-identity": [
    "ses_user"
  ],
  "myCustomTag1": [
    "myCustomTagValue1"
  ],
  "myCustomTag2": [
    "myCustomTagValue2"
  ],
},
"reject": {
  "reason": "Bad content"
}
}

Open record

The following is an example of an Open event record that Amazon SES publishes to Kinesis Data Firehose.

{
  "eventType": "Open",
  "mail": {
    "commonHeaders": {
      "from": [
        "sender@example.com"
      ],
      "messageId": "EXAMPLE7c191be45-e9aeb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
      "subject": "Message sent from Amazon SES",
      "to": [
        "recipient@example.com"
      ],
      "destination": [
        "recipient@example.com"
      ],
      "headers": [
        {
          "name": "X-SES-CONFIGURATION-SET",
          "value": "ConfigSet"
        },
        {
          "name": "X-SES-MESSAGE-TAGS",
          "value": "myCustomTag1=myCustomValue1, myCustomTag2=myCustomValue2"
        },
        {
          "name": "From",
          "value": "sender@example.com"
        },
        {
          "name": "To",
          "value": "recipient@example.com"
        }
      ]
    }
  }
}
Click record

The following is an example of a Click event record that Amazon SES publishes to Kinesis Data Firehose.

```json
{
  "eventType": "Click",
  "click": {
    "ipAddress": "192.0.2.1",
    "timestamp": "2017-08-09T22:00:19.652Z",
    "userAgent": "Mozilla/5.0 (iPhone; CPU iPhone OS 10_3_3 like Mac OS X) AppleWebKit/603.3.8 (KHTML, like Gecko) Mobile/14G60"
  }
}
```
"timestamp": "2017-08-09T23:51:25.570Z",
"userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/60.0.3112.90 Safari/537.36"
},
"mail": {
  "commonHeaders": {
    "from": ["sender@example.com"
  ],
  "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
  "subject": "Message sent from Amazon SES",
  "to": [
    "recipient@example.com"
  ]
},
"destination": ["recipient@example.com"],
"headers": [
  {"name": "X-SES-CONFIGURATION-SET", "value": "ConfigSet"},
  {"name": "X-SES-MESSAGE-TAGS", "value": "myCustomTag1=myCustomValue1, myCustomTag2=myCustomValue2"},
  {"name": "From", "value": "sender@example.com"},
  {"name": "To", "value": "recipient@example.com"},
  {"name": "Subject", "value": "Message sent from Amazon SES"},
  {"name": "MIME-Version", "value": "1.0"},
  {"name": "Content-Type", "value": "multipart/alternative; boundary="XBoundary""},
  {"name": "Message-ID", "value": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000"}
],
"headersTruncated": false,
"messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
"sendingAccountId": "123456789012",
"source": "sender@example.com",
"tags": {
  "myCustomTag1": ["myCustomValue1"],
  "myCustomTag2": ["myCustomValue2"]
},
"ses:caller-identity": ["ses_user"
Rendering Failure record

The following is an example of a Rendering Failure event record that Amazon SES publishes to Kinesis Data Firehose.

```json
{
  "eventType": "Rendering Failure",
  "mail": {
    "timestamp": "2018-01-22T18:43:06.197Z",
    "source": "sender@example.com",
    "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId": "123456789012",
    "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a0f8a-000000",
    "destination": [
      "recipient@example.com"
    ],
    "headersTruncated": false,
    "tags": {
      "ses:configuration-set": [
        "ConfigSet"
      ]
    },
    "failure": {
      "errorMessage": "Attribute 'attributeName' is not present in the rendering data.",
      "templateName": "MyTemplate"
    }
  }
}
```

DeliveryDelay record

The following is an example of a DeliveryDelay event record that Amazon SES publishes to Kinesis Data Firehose.

```json
{
  "eventType": "DeliveryDelay",
  "mail": {
    "timestamp": "2020-06-16T00:15:40.641Z",
    "source": "sender@example.com",
    "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId": "123456789012",
    "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a0f8a-000000",
    "destination": [
      "recipient@example.com"
    ],
    "headersTruncated": false,
    "tags": {
      "ses:configuration-set": [
        "ConfigSet"
      ]
    },
    "failure": {
      "errorMessage": "Message body could not be rendered.",
      "templateName": "MyTemplate"
    }
  }
}
```
Interpreting Amazon SES event data from Amazon SNS

Amazon SES publishes email sending events to Amazon Simple Notification Service (Amazon SNS) as JSON records. Amazon SNS then delivers notifications to the endpoints that are subscribed to the Amazon SNS topic associated with the event destination. For information about setting up topics and subscriptions in Amazon SNS, see Getting Started in the Amazon Simple Notification Service Developer Guide.

For a description of the record contents and for example records, see the following sections.

- Event record contents (p. 321)
- Event record examples (p. 330)

Contents of event data that Amazon SES publishes to Amazon SNS

Amazon SES publishes email sending event records to Amazon Simple Notification Service in JSON format.

The top-level JSON object contains an eventType string, a mail object, and either a Bounce, Complaint, Delivery, Send, Reject, Open, Click, Rendering Failure, or DeliveryDelay object, depending on the type of event.

You can find example records for all of these notification types in Examples of event data that Amazon SES publishes to Amazon SNS (p. 330).

Topics in this section:
- Top-level JSON object (p. 322)
- Mail object (p. 322)
- Bounce object (p. 324)
- Complaint object (p. 326)
- Delivery object (p. 327)
- Send object (p. 328)
- Reject object (p. 328)
- Open object (p. 328)
- Click object (p. 328)
- Rendering Failure object (p. 329)
- DeliveryDelay object (p. 329)
Top-level JSON object

The top-level JSON object in an email sending event record contains the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eventType</td>
<td>A string that describes the type of event. Possible values: Delivery, Send, Reject, Open, Click, Bounce, Complaint, Rendering Failure, or DeliveryDelay.</td>
</tr>
<tr>
<td>mail</td>
<td>A JSON object that contains information about the email that produced the event.</td>
</tr>
<tr>
<td>bounce</td>
<td>This field is only present if eventType is Bounce. It contains information about the bounce.</td>
</tr>
<tr>
<td>complaint</td>
<td>This field is only present if eventType is Complaint. It contains information about the complaint.</td>
</tr>
<tr>
<td>delivery</td>
<td>This field is only present if eventType is Delivery. It contains information about the delivery.</td>
</tr>
<tr>
<td>send</td>
<td>This field is only present if eventType is Send.</td>
</tr>
<tr>
<td>reject</td>
<td>This field is only present if eventType is Reject. It contains information about the rejection.</td>
</tr>
<tr>
<td>open</td>
<td>This field is only present if eventType is Open. It contains information about the open event.</td>
</tr>
<tr>
<td>click</td>
<td>This field is only present if eventType is Click. It contains information about the click event.</td>
</tr>
<tr>
<td>failure</td>
<td>This field is only present if eventType is Rendering Failure. It contains information about the rendering failure event.</td>
</tr>
<tr>
<td>deliveryDelay</td>
<td>This field is only present if eventType is DeliveryDelay. It contains information about the delayed delivery of an email.</td>
</tr>
</tbody>
</table>

Mail object

Each email sending event record contains information about the original email in the mail object. The JSON object that contains information about a mail object has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timestamp</td>
<td>The date and time, in ISO8601 format (YYYY-MM-DDThh:mm:ss.sZ), when the message was sent.</td>
</tr>
<tr>
<td>messageId</td>
<td>A unique ID that Amazon SES assigned to the message. Amazon SES returned this value to you when you sent the message.</td>
</tr>
</tbody>
</table>
### Field Name

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>source</strong></td>
<td>The email address that the message was sent from (the envelope MAIL FROM address).</td>
</tr>
<tr>
<td><strong>sourceArn</strong></td>
<td>The Amazon Resource Name (ARN) of the identity that was used to send the email. In the case of sending authorization, the sourceArn is the ARN of the identity that the identity owner authorized the delegate sender to use to send the email. For more information about sending authorization, see Using sending authorization (p. 149).</td>
</tr>
<tr>
<td><strong>sendingAccountId</strong></td>
<td>The AWS account ID of the account that was used to send the email. In the case of sending authorization, the sendingAccountId is the delegate sender's account ID.</td>
</tr>
<tr>
<td><strong>destination</strong></td>
<td>A list of email addresses that were recipients of the original mail.</td>
</tr>
<tr>
<td><strong>headersTruncated</strong></td>
<td>A string that specifies whether the headers are truncated in the notification, which occurs if the headers are larger than 10 KB. Possible values are true and false.</td>
</tr>
<tr>
<td><strong>headers</strong></td>
<td>A list of the email's original headers. Each header in the list has a name field and a value field.</td>
</tr>
<tr>
<td><strong>commonHeaders</strong></td>
<td>A list of the email's original, commonly used headers. Each header in the list has a name field and a value field.</td>
</tr>
</tbody>
</table>

**Note**
- This message ID was assigned by Amazon SES. You can find the message ID of the original email in the headers and commonHeaders fields of the mail object.
- Any message ID within the headers field is from the original message that you passed to Amazon SES. The message ID that Amazon SES subsequently assigned to the message is in the messageId field of the mail object.
- Any message ID within the commonHeaders field is from the original message that you passed to Amazon SES. The message ID that Amazon SES subsequently assigned to the message is in the messageId field of the mail object.
Bounce object

The JSON object that contains information about a Bounce event has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bounceType</td>
<td>The type of bounce, as determined by Amazon SES.</td>
</tr>
<tr>
<td>bounceSubType</td>
<td>The subtype of the bounce, as determined by Amazon SES.</td>
</tr>
<tr>
<td>bouncedRecipients</td>
<td>A list that contains information about the recipients of the original mail that bounced.</td>
</tr>
<tr>
<td>timestamp</td>
<td>The date and time, in ISO8601 format (YYYY-MM-DDThh:mm:ss.sZ), when the ISP sent the bounce notification.</td>
</tr>
<tr>
<td>feedbackId</td>
<td>A unique ID for the bounce.</td>
</tr>
<tr>
<td>reportingMTA</td>
<td>The value of the Reporting-MTA field from the DSN. This is the value of the Message Transfer Authority (MTA) that attempted to perform the delivery, relay, or gateway operation described in the DSN.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> This field only appears if a delivery status notification (DSN) was attached to the bounce.</td>
</tr>
</tbody>
</table>

Bounced recipients

A bounce event may pertain to a single recipient or to multiple recipients. The bouncedRecipients field holds a list of objects—one object per recipient whose email address produced a bounce—and contains the following field.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>emailAddress</td>
<td>The email address of the recipient. If a DSN is available, this is the value of the Final-Recipient field from the DSN.</td>
</tr>
</tbody>
</table>

Optionally, if a DSN is attached to the bounce, the following fields may also be present.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>The value of the Action field from the DSN. This indicates the action performed by the reporting MTA as a result of its attempt to deliver the message to this recipient.</td>
</tr>
</tbody>
</table>
| status     | The value of the Status field from the DSN. This is the per-recipient transport-independent
Field Name | Description
---|---
status code that indicates the delivery status of the message.
diagnosticCode | The status code issued by the reporting MTA. This is the value of the Diagnostic-Code field from the DSN. This field may be absent in the DSN (and therefore also absent in the JSON).

### Bounce types

Each bounce event is of one of the types shown in the following table.

The event publishing system only publishes hard bounces and soft bounces that are no longer retried by Amazon SES. When you receive bounces marked Permanent, you should remove the corresponding email addresses from your mailing list; you will not be able to send to them in the future. Transient bounces are sent to you when a message has soft bounced several times, and Amazon SES has stopped trying to re-deliver it. You may be able to successfully resend to an address that initially resulted in a Transient bounce in the future.

<table>
<thead>
<tr>
<th>bounceType</th>
<th>bounceSubType</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Undetermined</td>
<td>Undetermined</td>
<td>Amazon SES was unable to determine a specific bounce reason.</td>
</tr>
<tr>
<td>Permanent</td>
<td>General</td>
<td>Amazon SES received a general hard bounce. If you receive this type of bounce, you should remove the recipient's email address from your mailing list.</td>
</tr>
<tr>
<td>Permanent</td>
<td>NoEmail</td>
<td>Amazon SES received a permanent hard bounce because the target email address does not exist. If you receive this type of bounce, you should remove the recipient's email address from your mailing list.</td>
</tr>
<tr>
<td>Permanent</td>
<td>Suppressed</td>
<td>Amazon SES has suppressed sending to this address because it has a recent history of bouncing as an invalid address. For information about how to remove an address from the suppression list, see Using the Amazon SES global suppression list (p. 192).</td>
</tr>
<tr>
<td>Permanent</td>
<td>OnAccountSuppressionList</td>
<td>Amazon SES has suppressed sending to this address because it is on the account-level suppression list (p. 184).</td>
</tr>
<tr>
<td>Transient</td>
<td>General</td>
<td>Amazon SES received a general bounce. You may be able to successfully send to this recipient in the future.</td>
</tr>
<tr>
<td>Transient</td>
<td>MailboxFull</td>
<td>Amazon SES received a mailbox full bounce. You may be able to successfully send to this recipient in the future.</td>
</tr>
</tbody>
</table>
### bounceType | bounceSubType | Description
--- | --- | ---
Transient | MessageTooLarge | Amazon SES received a message too large bounce. You may be able to successfully send to this recipient if you reduce the size of the message.
Transient | ContentRejected | Amazon SES received a content rejected bounce. You may be able to successfully send to this recipient if you change the content of the message.
Transient | AttachmentRejected | Amazon SES received an attachment rejected bounce. You may be able to successfully send to this recipient if you remove or change the attachment.

### Complaint object

The JSON object that contains information about a complaint event has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>complainedRecipients</td>
<td>A list that contains information about recipients that may have submitted the complaint.</td>
</tr>
<tr>
<td>timestamp</td>
<td>The date and time, in ISO8601 format (YYYY-MM-DDThh:mm:ss.sZ), when the ISP sent the complaint notification.</td>
</tr>
<tr>
<td>feedbackId</td>
<td>A unique ID for the complaint.</td>
</tr>
<tr>
<td>complaintSubType</td>
<td>The subtype of the complaint, as determined by Amazon SES.</td>
</tr>
</tbody>
</table>

Further, if a feedback report is attached to the complaint, the following fields may be present.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userAgent</td>
<td>The value of the User-Agent field from the feedback report. This indicates the name and version of the system that generated the report.</td>
</tr>
<tr>
<td>complaintFeedbackType</td>
<td>The value of the Feedback-Type field from the feedback report received from the ISP. This contains the type of feedback.</td>
</tr>
<tr>
<td>arrivalDate</td>
<td>The value of the Arrival-Date or Received-Date field from the feedback report in ISO8601 format (YYYY-MM-DDThh:mm:ss.sZ). This field may be absent in the report (and therefore also absent in the JSON).</td>
</tr>
</tbody>
</table>

### Complained recipients

The complainedRecipients field contains a list of recipients that may have submitted the complaint.
Important
Most ISPs redact the email addresses of recipients who submit complaints. For this reason, the `complainedRecipients` field includes a list of everyone who was sent the email whose address is on the domain that issued the complaint notification.

JSON objects in this list contain the following field.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>emailAddress</td>
<td>The email address of the recipient.</td>
</tr>
</tbody>
</table>

Complaint types

You may see the following complaint types in the `complaintFeedbackType` field as assigned by the reporting ISP, according to the Internet Assigned Numbers Authority website:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>abuse</td>
<td>Indicates unsolicited email or some other kind of email abuse.</td>
</tr>
<tr>
<td>auth-failure</td>
<td>Email authentication failure report.</td>
</tr>
<tr>
<td>fraud</td>
<td>Indicates some kind of fraud or phishing activity.</td>
</tr>
<tr>
<td>not-spam</td>
<td>Indicates that the entity providing the report does not consider the message to be spam. This may be used to correct a message that was incorrectly tagged or categorized as spam.</td>
</tr>
<tr>
<td>other</td>
<td>Indicates any other feedback that does not fit into other registered types.</td>
</tr>
<tr>
<td>virus</td>
<td>Reports that a virus is found in the originating message.</td>
</tr>
</tbody>
</table>

Complaint subtypes

The value of the `complaintSubType` field can either be null or `OnAccountSuppressionList`. If the value is `OnAccountSuppressionList`, Amazon SES accepted the message, but didn't attempt to send it because it was on the account-level suppression list (p. 184).

Delivery object

The JSON object that contains information about a Delivery event has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timestamp</td>
<td>The date and time when Amazon SES delivered the email to the recipient's mail server, in ISO8601 format (<code>YYYY-MM-DDThh:mm:ss.sZ</code>).</td>
</tr>
<tr>
<td>processingTimeMillis</td>
<td>The time in milliseconds between when Amazon SES accepted the request from the sender to</td>
</tr>
</tbody>
</table>
### Field Name | Description
--- | ---
when Amazon SES passed the message to the recipient's mail server.  
recipients | A list of intended recipients that the delivery event applies to.  
smtResponse | The SMTP response message of the remote ISP that accepted the email from Amazon SES. This message will vary by email, by receiving mail server, and by receiving ISP.  
reportingMTA | The host name of the Amazon SES mail server that sent the mail.

**Send object**

The JSON object that contains information about a send event is always empty.

**Reject object**

The JSON object that contains information about a reject event has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reason</td>
<td>The reason the email was rejected. The only possible value is <strong>Bad content</strong>, which means that Amazon SES detected that the email contained a virus. When a message is rejected, Amazon SES stops processing it, and doesn't attempt to deliver it to the recipient's mail server.</td>
</tr>
</tbody>
</table>

**Open object**

The JSON object that contains information about a open event has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ipAddress</td>
<td>The recipient's IP address.</td>
</tr>
<tr>
<td>timestamp</td>
<td>The date and time when the open event occurred in ISO8601 format (<strong>YYYY-MM-DDThh:mm:ss.sZ</strong>).</td>
</tr>
<tr>
<td>userAgent</td>
<td>The user agent of the device or email client that the recipient used to open the email.</td>
</tr>
</tbody>
</table>

**Click object**

The JSON object that contains information about a click event has the following fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ipAddress</td>
<td>The recipient's IP address.</td>
</tr>
</tbody>
</table>
Field Name | Description
--- | ---
timestamp | The date and time when the click event occurred in ISO8601 format (`YYYY-MM-DDThh:mm:ss.sZ`).
userAgent | The user agent of the client that the recipient used to click a link in the email.
link | The URL of the link that the recipient clicked.
linkTags | A list of tags that were added to the link using the `ses:tags` attribute. For more information about adding tags to links in your emails, see Q5. Can I tag links with unique identifiers? (p. 508) in the Amazon SES email sending metrics FAQs (p. 506).

**Rendering Failure object**

The JSON object that contains information about a Rendering Failure event has the following fields.

Field Name | Description
--- | ---
templateName | The name of the template used to send the email.
errorMessage | A message that provides more information about the rendering failure.

**DeliveryDelay object**

The JSON object that contains information about a DeliveryDelay event has the following fields.

Field Name | Description
--- | ---
delayType | The type of delay. Possible values are:
  - **InternalFailure** – An internal Amazon SES issue caused the message to be delayed.
  - **General** – A generic failure occurred during the SMTP conversation.
  - **MailboxFull** – The recipient's mailbox is full and is unable to receive additional messages.
  - **SpamDetected** – The recipient's mail server has detected a large amount of unsolicited email from your account.
  - **RecipientServerError** – A temporary issue with the recipient's email server is preventing the delivery of the message.
  - **IPFailure** – The IP address that's sending the message is being blocked or throttled by the recipient's email provider.
  - **TransientCommunicationGeneral** – There was a temporary communication failure during the
Field Name | Description
--- | ---
SMTP conversation with the recipient’s email provider.
- **BYOIPHostNameLookupUnavailable** – Amazon SES was unable to look up the DNS hostname for your IP addresses. This type of delay only occurs when you use Bring Your Own IP (p. 180).
- **Undetermined** – Amazon SES wasn’t able to determine the reason for the delivery delay.
delayedRecipients | An object that contains information about the recipient of the email.
expirationTime | The date and time when Amazon SES will stop trying to deliver the message. This value is shown in ISO 8601 format.
reportingMTA | The IP address of the Message Transfer Agent (MTA) that reported the delay.
timestamp | The date and time when the delay occurred, shown in ISO 8601 format.

Delayed recipients

The `delayedRecipients` object contains the following values.

Field Name | Description
--- | ---
emailAddress | The email address that resulted in the delivery of the message being delayed.
status | The SMTP status code associated with the delivery delay.
diagnosticCode | The diagnostic code provided by the receiving Message Transfer Agent (MTA).

Examples of event data that Amazon SES publishes to Amazon SNS

This section provides examples of the types of email sending event records that Amazon SES publishes to Amazon SNS.

Topics in this section:
- Bounce record (p. 331)
- Complaint record (p. 332)
- Delivery record (p. 333)
- Send record (p. 334)
- Reject record (p. 336)
- Open record (p. 337)
- Click record (p. 338)
- Rendering Failure record (p. 340)
Bounce record

The following is an example of a Bounce event record that Amazon SES publishes to Amazon SNS.

```json
{
    "eventType": "Bounce",
    "bounce": {
        "bounceType": "Permanent",
        "bounceSubType": "General",
        "bouncedRecipients": [
            {
                "emailAddress": "recipient@example.com",
                "action": "failed",
                "status": "5.1.1",
                "diagnosticCode": "smtp; 550 5.1.1 user unknown"
            }
        ],
        "timestamp": "2017-08-05T00:41:02.669Z",
        "feedbackId": "01000157c44f053b-61b59c11-9236-11e6-8f96-7be8aexample-000000",
        "reportingMTA": "dsn; mta.example.com"
    },
    "mail": {
        "timestamp": "2017-08-05T00:40:02.012Z",
        "source": "Sender Name <sender@example.com>",
        "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
        "sendingAccountId": "123456789012",
        "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
        "destination": [
            "recipient@example.com"
        ],
        "headersTruncated": false,
        "headers": [
            {
                "name": "From",
                "value": "Sender Name <sender@example.com>"
            },
            {
                "name": "To",
                "value": "recipient@example.com"
            },
            {
                "name": "Subject",
                "value": "Message sent from Amazon SES"
            },
            {
                "name": "MIME-Version",
                "value": "1.0"
            },
            {
                "name": "Content-Type",
                "value": "multipart/alternative; boundary="_Part_7307378_1629847660.1516840721503"
            }
        ],
        "commonHeaders": {
            "from": [
                "Sender Name <sender@example.com>"
            ],
            "to": [
                "recipient@example.com"
            ],
            "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000"
        }
    }
}
```
Complaint record

The following is an example of a Complaint event record that Amazon SES publishes to Amazon SNS.

```json
{
  "eventType":"Complaint",
  "complaint": {
    "complainedRecipients": [
      { "emailAddress":"recipient@example.com" }
    ],
    "timestamp":"2017-08-05T00:41:02.669Z",
    "feedbackId":"01000157c44f053b-61b59c11-9236-11e6-8f96-7be8aexample-000000",
    "userAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/60.0.3112.90 Safari/537.36",
    "complaintFeedbackType":"abuse",
    "arrivalDate":"2017-08-05T00:41:02.669Z"
  },
  "mail": {
    "timestamp":"2017-08-05T00:40:01.123Z",
    "source":"Sender Name <sender@example.com>",
    "sourceArn":"arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId":"123456789012",
    "messageId":"EXAMPLE7c191be45-e9ed9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination": ["recipient@example.com"],
    "headersTruncated":false,
    "headers": [
      { "name":"From", "value":"Sender Name <sender@example.com>" },
      { "name":"To", "value":"recipient@example.com" },
      { "name":"Subject", "value":"Message sent from Amazon SES" },
      { "name":"MIME-Version", "value":"1.0" }
    ]
  }
}
```
Delivery record

The following is an example of a Delivery event record that Amazon SES publishes to Amazon SNS.
Send record

The following is an example of a Send event record that Amazon SES publishes to Amazon SNS.

```json
{
    "eventType": "Send",
    "mail": {
        "timestamp": "2016-10-14T05:02:16.645Z",
        "from": ["sender@example.com"],
        "to": ["recipient@example.com"],
        "messageId": "EXAMPLE7c191be45-e9aedd9a-02f9-4d12-a87d-dd0099a07f8a-000000",
        "subject": "Message sent from Amazon SES",
        "commonHeaders": {
            "from": ["sender@example.com"],
            "to": ["recipient@example.com"],
            "messageId": "EXAMPLE7c191be45-e9aedd9a-02f9-4d12-a87d-dd0099a07f8a-000000",
            "subject": "Message sent from Amazon SES"
        },
        "tags": {
            "ses:configuration-set": ["ConfigSet"],
            "ses:source-ip": ["192.0.2.0"],
            "ses:from-domain": ["example.com"],
            "ses:caller-identity": ["ses_user"],
            "ses:outgoing-ip": ["192.0.2.0"],
            "myCustomTag1": ["myCustomTagValue1"],
            "myCustomTag2": ["myCustomTagValue2"
        }
    }
}
"source": "sender@example.com",
"sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
"sendingAccountId": "123456789012",
"messageId": "EXAMPLE7c191be45-e9ae0b9a-02f9-4d12-a87d-dd0099a07f8a-000000",
"destination": [
  "recipient@example.com"
],
"headersTruncated": false,
"headers": [
  {
    "name": "From",
    "value": "sender@example.com"
  },
  {
    "name": "To",
    "value": "recipient@example.com"
  },
  {
    "name": "Subject",
    "value": "Message sent from Amazon SES"
  },
  {
    "name": "MIME-Version",
    "value": "1.0"
  },
  {
    "name": "Content-Type",
    "value": "multipart/mixed; boundary="----=_Part_0_716996660.1476423363411"
  },
  {
    "name": "X-SES-MESSAGE-TAGS",
    "value": "myCustomTag1=myCustomTagValue1, myCustomTag2=myCustomTagValue2"
  }
],
"commonHeaders": {
  "from": [
    "sender@example.com"
  ],
  "to": [
    "recipient@example.com"
  ],
  "messageId": "EXAMPLE7c191be45-e9ae0b9a-02f9-4d12-a87d-dd0099a07f8a-000000",
  "subject": "Message sent from Amazon SES"
},
"tags": {
  "ses:configuration-set": [ "ConfigSet" ],
  "ses:source-ip": [ "192.0.2.0" ],
  "ses:from-domain": [ "example.com" ],
  "ses:caller-identity": [ "ses_user" ],
  "myCustomTag1": [ "myCustomTagValue1" ],
  "myCustomTag2": [ "myCustomTagValue2" ]
},
"send": {"}
Reject record

The following is an example of a `Reject` event record that Amazon SES publishes to Amazon SNS.

```json
{
    "eventType": "Reject",
    "mail": {
        "timestamp": "2016-10-14T17:38:15.211Z",
        "source": "sender@example.com",
        "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
        "sendingAccountId": "123456789012",
        "messageId": "EXAMPLE7c191be45-e9aeb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
        "destination": [
            "sender@example.com"
        ],
        "headersTruncated": false,
        "headers": [
            {
                "name": "From",
                "value": "sender@example.com"
            },
            {
                "name": "To",
                "value": "recipient@example.com"
            },
            {
                "name": "Subject",
                "value": "Message sent from Amazon SES"
            },
            {
                "name": "MIME-Version",
                "value": "1.0"
            },
            {
                "name": "Content-Type",
                "value": "multipart/mixed; boundary="qMm9M+Fa2AknNoGS""
            },
            {
                "name": "X-SES-MESSAGE-TAGS",
                "value": "myCustomTag1=myCustomTagValue1, myCustomTag2=myCustomTagValue2"
            }
        ],
        "commonHeaders": {
            "from": [
                "sender@example.com"
            ],
            "to": [
                "recipient@example.com"
            ],
            "messageId": "EXAMPLE7c191be45-e9aeb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
            "subject": "Message sent from Amazon SES"
        },
        "tags": {
            "ses:configuration-set": [
                "ConfigSet"
            ],
            "ses:source-ip": [
                "192.0.2.0"
            ],
            "ses:from-domain": [
                "example.com"
            ]
        }
    }
}
```
Open record

The following is an example of an Open event record that Amazon SES publishes to Amazon SNS.

```json
{
  "eventType": "Open",
  "mail": {
    "commonHeaders": {
      "from": [
        "sender@example.com"
      ],
      "messageId": "EXAMPLE7c191be45-e9aeb9a-02f9-4d12-a87d-dd099a07f8a-000000",
      "subject": "Message sent from Amazon SES",
      "to": [
        "recipient@example.com"
      ],
      "destination": [
        "recipient@example.com"
      ],
      "headers": [
        {
          "name": "X-SES-CONFIGURATION-SET",
          "value": "ConfigSet"
        },
        {
          "name": "X-SES-MESSAGE-TAGS",
          "value": "myCustomTag1=myCustomValue1, myCustomTag2=myCustomValue2"
        },
        {
          "name": "From",
          "value": "sender@example.com"
        },
        {
          "name": "To",
          "value": "recipient@example.com"
        },
        {
          "name": "Subject",
          "value": "Message sent from Amazon SES"
        },
        {
          "name": "MIME-Version",
          "value": "1.0"
        },
        {
          "name": "Content-Type",
          "value": "multipart/alternative; boundary="XBoundary""
        }
      ]
    }
  },
  "reject": {
    "reason": "Bad content"
  }
}
```
Click record

The following is an example of a Click event record that Amazon SES publishes to Amazon SNS.

```json
{
    "eventType": "Click",
    "click": {
        "ipAddress": "192.0.2.1",
        "linkTags": {
            "samplekey0": ["samplevalue0"],
            "samplekey1": ["samplevalue1"
        ],
        "timestamp": "2017-08-09T23:51:25.570Z",
        "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/60.0.3112.90 Safari/537.36"
    },
    "mail": {
        "commonHeaders": {
            "from": [
                "sender@example.com"
            ],
            "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
            "subject": "Message sent from Amazon SES",
            "sendingAccountId": "123456789012",
            "source": "sender@example.com",
            "tags": {
                "myCustomTag1": ["myCustomValue1"],
                "myCustomTag2": ["myCustomValue2"],
                "ses:caller-identity": ["ses-user"],
                "ses:configuration-set": ["ConfigSet"],
                "ses:from-domain": ["example.com"],
                "ses:source-ip": ["192.0.2.0"
            ],
            "timestamp": "2017-08-09T21:59:49.927Z"
        },
        "open": {
            "ipAddress": "192.0.2.1",
            "timestamp": "2017-08-09T22:00:19.652Z",
            "userAgent": "Mozilla/5.0 (iPhone; CPU iPhone OS 10_3_3 like Mac OS X) AppleWebKit/603.3.8 (KHTML, like Gecko) Mobile/14G60"
        }
    }
}
```
"to": [  "recipient@example.com"
],
"destination": [  "recipient@example.com"
],
"headers": [  {  "name": "X-SES-CONFIGURATION-SET",
      "value": "ConfigSet"
  },  {  "name": "X-SES-MESSAGE-TAGS",
      "value": "myCustomTag1=myCustomValue1, myCustomTag2=myCustomValue2"
  },  {  "name": "From",
      "value": "sender@example.com"
  },  {  "name": "To",
      "value": "recipient@example.com"
  },  {  "name": "Subject",
      "value": "Message sent from Amazon SES"
  },  {  "name": "MIME-Version",
      "value": "1.0"
  },  {  "name": "Content-Type",
      "value": "multipart/alternative; boundary="XBoundary"
  },  {  "name": "Message-ID",
      "value": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000"
  }],
"headersTruncated": false,
"messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
"sendingAccountId": "123456789012",
"source": "sender@example.com",
"tags": {
  "myCustomTag1": [  "myCustomValue1"
  ],
  "myCustomTag2": [  "myCustomValue2"
  ],
  "ses:caller-identity": [  "ses_user"
  ],
  "ses:configuration-set": [  "ConfigSet"
  ],
  "ses:from-domain": [  "example.com"
  ],
  "ses:source-ip": [  "192.0.2.0"
  ],
  "timestamp": "2017-08-09T23:50:05.795Z"
Rendering Failure record

The following is an example of a Rendering Failure event record that Amazon SES publishes to Amazon SNS.

```
{
  "eventType": "Rendering Failure",
  "mail": {
    "timestamp": "2018-01-22T18:43:06.197Z",
    "source": "sender@example.com",
    "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId": "123456789012",
    "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination": ["recipient@example.com"],
    "headersTruncated": false,
    "tags": {
      "ses:configuration-set": ["ConfigSet"]
    }
  },
  "failure": {
    "errorMessage": "Attribute 'attributeName' is not present in the rendering data.",
    "templateName": "MyTemplate"
  }
}
```

DeliveryDelay record

The following is an example of a DeliveryDelay event record that Amazon SES publishes to Amazon SNS.

```
{
  "eventType": "DeliveryDelay",
  "mail": {
    "timestamp": "2020-06-16T00:15:40.641Z",
    "source": "sender@example.com",
    "sourceArn": "arn:aws:ses:us-east-1:123456789012:identity/sender@example.com",
    "sendingAccountId": "123456789012",
    "messageId": "EXAMPLE7c191be45-e9aedb9a-02f9-4d12-a87d-dd0099a07f8a-000000",
    "destination": ["recipient@example.com"],
    "headersTruncated": false,
    "tags": {
      "ses:configuration-set": ["ConfigSet"]
    }
  },
  "deliveryDelay": {
    "timestamp": "2020-06-16T00:25:40.095Z",
    "delayType": "TransientCommunicationFailure",
    "expirationTime": "2020-06-16T00:25:40.914Z",
    "delayedRecipients": [{
      "emailAddress": "recipient@example.com",
      "status": "4.4.1",
      "diagnosticCode": "smtp: 421 4.4.1 Unable to connect to remote host"
    }]
  }
}
```
Event publishing tutorials

This section provides tutorials that demonstrate how to use Amazon SES event publishing with AWS services that enable you to analyze and visualize your data.

Topics in this section:
- Analyze email sending events with Amazon Redshift (p. 341)
- Graph email sending events in Amazon CloudWatch (p. 352)
- Analyze email sending events with Amazon Kinesis Data Analytics (p. 355)

Analyze email sending events with Amazon Redshift

In this tutorial, you publish Amazon SES email sending events to an Amazon Kinesis Data Firehose delivery stream that publishes data to Amazon Redshift. You then connect to the Amazon Redshift database and use a SQL query tool to query the database for Amazon SES email sending events that meet certain criteria.

The following sections walk you through the process.

- Prerequisites (p. 341)
- Step 1: Create an Amazon Redshift Cluster (p. 342)
- Step 2: Connect to Your Amazon Redshift Cluster (p. 342)
- Step 3: Create a Database Table (p. 345)
- Step 4: Create a Kinesis Data Firehose Delivery Stream (p. 347)
- Step 5: Set up a Configuration Set (p. 350)
- Step 6: Send Emails (p. 350)
- Step 7: Query Email Sending Events (p. 351)

Prerequisites

For this tutorial, you will need the following:

- An AWS account – To access any web service that AWS offers, you must first create an AWS account at https://aws.amazon.com/.
- Verified email address – To send emails using Amazon SES, you must verify your "From" address or domain to show that you own it. If you are in the sandbox, you also must verify your "To" addresses. You can verify email addresses or entire domains, but this tutorial requires a verified email address so that you can send an email from the Amazon SES console, which is the simplest way to send an email. For information about how to verify an email address, see Verifying email addresses in Amazon SES (p. 47).
- A SQL query tool – Amazon Redshift does not provide or install any SQL client tools or libraries, so you must install one that you can use to access the Amazon Redshift clusters that contain your Amazon SES events. In this tutorial, we use SQL Workbench/J, a free, DBMS-independent, cross-platform SQL query tool. This section includes procedures for installing SQL Workbench/J.

To install SQL Workbench/J

1. Review the SQL Workbench/J software license.
2. Go to the SQL Workbench/J website and download the appropriate package for your operating system.

3. Go to Installing and starting SQL Workbench/J and install SQL Workbench/J.

   **Important**
   
   Note the Java runtime version prerequisites for SQL Workbench/J and ensure you are using that version. Otherwise, this client application will not run.

4. Go to Configure a JDBC Connection and download an Amazon Redshift JDBC driver to enable SQL Workbench/J to connect to your cluster.

Next Step

**Step 1: Create an Amazon Redshift Cluster (p. 342)**

**Step 1: Create an Amazon Redshift Cluster**

To create an Amazon Redshift cluster, go the Amazon Redshift console and choose Launch Cluster. A wizard guides you through choosing options for your cluster, and it provides default values for most options.

For this simple tutorial, type a cluster name and password, and then you can use all of the default values. You do not need to set any values specific to Amazon SES event publishing.

   **Important**

   The cluster that you deploy for this tutorial will run in a live environment. As long as it is running, it will accrue charges to your AWS account. To avoid unnecessary charges, you should delete your cluster when you are done with it. For pricing information, go to the Amazon Redshift pricing page.

Next Step

**Step 2: Connect to Your Amazon Redshift Cluster (p. 342)**

**Step 2: Connect to Your Amazon Redshift Cluster**

Now you will connect to your cluster by using a SQL client tool. For this tutorial, you use the SQL Workbench/J client that you installed in the prerequisites section (p. 341).

Complete this section by performing the following steps:

- Getting Your Connection String (p. 342)
- Connecting to Your Cluster From SQL Workbench/J (p. 343)

**Getting Your Connection String**

The following procedure shows how to get the connection string that you will need to connect to your Amazon Redshift cluster from SQL Workbench/J.

**To get your connection string**

1. In the Amazon Redshift console, in the navigation pane, choose Clusters.
2. To open your cluster, choose your cluster name.
3. On the Configuration tab, under Cluster Database Properties, copy the JDBC URL of the cluster.

   **Note**

   The endpoint for your cluster is not available until the cluster is created and in the available state.
Connecting to Your Cluster From SQL Workbench/J

The following procedure shows how to connect to your cluster from SQL Workbench/J. This procedure assumes that you installed SQL Workbench/J on your computer as described in Prerequisites (p. 341).

To connect to your cluster from SQL Workbench/J

1. Open SQL Workbench/J.
2. Choose File, and then choose Connect window.
3. Choose the Create a new connection profile button.
4. In the New profile text box, type a name for the profile.
5. At the bottom of the window, on the left, choose Manage Drivers.
6. In the Manage Drivers dialog box, choose the Create a new entry button, and then add the driver as follows.
a. In the **Name** box, type a name for the driver.
b. Next to **Library**, choose the folder icon.
c. Navigate to the location of the driver you downloaded in Configure a JDBC Connection, select the driver, and then choose **Open**.
d. Choose **OK**.

You will be taken back to the **Select Connection Profile** dialog box.

7. For **Driver**, choose the driver that you just added.
8. For **URL**, paste the JDBC URL that you copied from the Amazon Redshift console.
9. For **Username**, type the username that you chose when you set up the Amazon Redshift cluster (p. 342).
10. For **Password**, type the password that you chose when you set up the Amazon Redshift cluster.
11. Select **Autocommit**.
12. To test the connection, choose **Test**.

**Note**

If the connection attempt times out, you might need to add your IP address to the security group that allows incoming traffic from IP addresses. For more information, see The Connection Is Refused or Fails in the *Amazon Redshift Database Developer Guide*. 
13. On the top menu bar, choose the **Save profile list** button.
14. Choose **OK**.

SQL Workbench/J will connect to your Amazon Redshift cluster.

**Next Step**

**Step 3: Create a Database Table (p. 345)**

**Step 3: Create a Database Table**

After you connect to the initial database in Amazon Redshift, you typically use the initial database as the base for creating a new database. However, in this simple tutorial, we create a table to hold your Amazon SES event publishing data directly within the initial database.

For this tutorial, let's assume that we're interested in the following fields within the email sending event records (p. 301). All of these fields, except for `mail.tags.campaign`, are provided automatically by Amazon SES. We introduce the `mail.tags.campaign` field when we send an email using `campaign` as a message tag in **Step 6: Send Emails (p. 350)**.

- `mail.messageId`
- `eventType`
- `mail.sendingAccountId`
- `mail.timestamp`
- `mail.destination`
- `mail.tags.ses:configuration-set`
- `mail.tags.campaign`
To access this information within your database, you must create a table. The following procedure shows how to specify this information when you create the table in your database.

**Note**

We assume that SQL Workbench/J is currently open on your computer, and it is connected to your Amazon Redshift cluster, as described in previous step (p. 342).

**To create a table using SQL Workbench/J**

1. In SQL Workbench/J, copy the following code and paste it into the **Statement 1** window.

```sql
create table ses (
  message_id varchar(200) not null,
  event_type varchar(20) not null,
  sending_account_id char(12),
  timestamp varchar(50),
  destination text,
  configuration_set text,
  campaign text
);
```

2. Place the cursor within the statement (somewhere before the semicolon), and then choose the **Execute current statement** button, as shown in the following figure.

3. In the **Messages** pane, verify that your table was successfully created.

Next Step

**Step 4: Create a Kinesis Data Firehose Delivery Stream (p. 347)**
Step 4: Create a Kinesis Data Firehose Delivery Stream

To publish email sending events to Amazon Kinesis Data Firehose, you must create a Kinesis Data Firehose delivery stream. When you set up a Kinesis Data Firehose delivery stream, you choose where Kinesis Data Firehose publishes the data. For this tutorial, we will set up Kinesis Data Firehose to publish the data to Amazon Redshift, and choose to have Kinesis Data Firehose publish the records to Amazon S3 as an intermediary step. In the process, we need to specify how Amazon Redshift should copy records from Amazon S3 into the table we created in the previous step (p. 345).

This section shows how to create a Kinesis Data Firehose delivery stream that sends data to Amazon Redshift, and how to edit the delivery stream to specify how Amazon Redshift should copy the Amazon SES event publishing data to Amazon S3.

Note
You must have already set up the Amazon Redshift cluster (p. 342), connected to your cluster (p. 342), and created a database table (p. 345), as explained previous steps.

Creating a Kinesis Data Firehose Delivery Stream

The following procedure shows how to create a Kinesis Data Firehose delivery stream that publishes data to Amazon Redshift, using Amazon S3 as the intermediary data location.

To create a delivery stream from Kinesis Data Firehose to Amazon Redshift

1. Sign in to the AWS Management Console and open the Kinesis Data Firehose console at https://console.aws.amazon.com/firehose/.
2. Choose Create Delivery Stream.
3. On the Destination page, choose the following options.
   - Destination – Choose Amazon Redshift.
   - Delivery stream name – Type a name for the delivery stream.
   - S3 bucket – Choose New S3 bucket, type a bucket name, choose the region, and then choose Create Bucket.
   - Redshift cluster – Choose the Amazon Redshift cluster that you created in a previous step.
   - Redshift database – Type dev, which is the default database name.
   - Redshift table – Type ses, which is the table you created in Step 3: Create a Database Table (p. 345).
   - Redshift table columns – Leave this field empty.
   - Redshift username – Type the username that you chose when you set up the Amazon Redshift cluster (p. 342).
   - Redshift password – Type the password that you chose when you set up the Amazon Redshift cluster.
   - Redshift COPY options – Leave this field empty.
   - Retry duration – Leave this at its default value.
   - COPY command – Leave this at its default value. You will update it in the next procedure.
4. Choose Next.
5. On the Configuration page, leave the fields at the default settings for this simple tutorial. The only step you must do is select an IAM role that enables Kinesis Data Firehose to access your resources, as explained in the following procedure.
   a. For IAM Role, choose Select an IAM role.
   b. In the drop-down menu, under Create/Update existing IAM role, choose Firehose delivery IAM role.
You will be taken to the IAM console.

c. In the IAM console, leave the fields at their default settings, and then choose Allow.

You will return to the Kinesis Data Firehose delivery stream set-up steps in the Kinesis Data Firehose console.

6. Choose Next.

7. On the Review page, review your settings, and then choose Create Delivery Stream.

Setting Amazon Redshift Copy Options

Next, you must specify to Amazon Redshift how to copy the Amazon SES event publishing JSON records into the database table you created in Step 3: Create a Database Table (p. 345). You do this by editing the copy options in the Kinesis Data Firehose delivery stream.

For this procedure, you must create a JSONPaths file. A JSONPaths file is a text file that specifies to the Amazon Redshift COPY command how to parse the JSON source data. We provide a JSONPaths file in the procedure. For more information about JSONPaths files, see COPY from JSON Format in the Amazon Redshift Database Developer Guide.

You upload the JSONPaths file to the Amazon S3 bucket you set up when you created the Kinesis Data Firehose delivery stream, and then edit the COPY options of the Kinesis Data Firehose delivery stream to use the JSONPaths file you uploaded. These steps are explained in the following procedure.

To set Amazon Redshift COPY command options

1. Create a JSONPaths file – On your computer, create a file called jsonpaths.json. Copy the following text into the file, and then save the file.

   ```json
   { "jsonpaths": [ 
   ```
2. **Upload the JSONPaths file to the Amazon S3 bucket** – Go to the Amazon S3 console and upload the file to the bucket you created when you set up the Kinesis Data Firehose delivery stream in *Creating a Kinesis Data Firehose Delivery Stream (p. 347).*

3. **Set the COPY command in the Kinesis Data Firehose delivery stream settings** – Now you have the information you need to set the syntax of the COPY command that Amazon Redshift uses when it puts your data in the table you created. The following procedure shows how to update the COPY command information in the Kinesis Data Firehose delivery stream settings.

1. Go to the Kinesis Data Firehose console.
2. Under **Redshift Delivery Streams**, choose the Kinesis Data Firehose delivery stream that you created for Amazon SES event publishing.
3. On the **Details** page, choose **Edit**.
4. In the **Redshift COPY options** box, type the following text, replacing the following values with your own values:
   - **S3-BUCKET-NAME** – The name of the Amazon S3 bucket where Kinesis Data Firehose places your data for Amazon Redshift to access. You created this bucket when you set up your Kinesis Data Firehose delivery stream in *Step 4: Create a Kinesis Data Firehose Delivery Stream (p. 347).* An example is `my-bucket`.
   - **REGION** – The region in which your Amazon SES, Kinesis Data Firehose, Amazon S3, and Amazon Redshift resources are located. An example is `us-west-2`.

   ```
   json 's3://S3-BUCKET-NAME/jsonpaths.json' region 'REGION';
   ```

5. Choose **Save**.
Next Step

Step 5: Set up a Configuration Set (p. 350)

Step 5: Set up a Configuration Set

To set up Amazon SES to publish your email sending events to Amazon Kinesis Data Firehose, you first create a configuration set, and then you add a Kinesis Data Firehose event destination to the configuration set. This section shows how to accomplish those tasks.

If you already have a configuration set, you can add a Kinesis Data Firehose destination to your existing configuration set. In this case, skip to Adding a Kinesis Data Firehose Event Destination (p. 350).

Creating a Configuration Set

The following procedure shows how to create a configuration set.

**To create a configuration set**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, choose **Configuration Sets**.
3. In the content pane, choose **Create Configuration Set**.
4. Type a name for the configuration set, and then choose **Create Configuration Set**.
5. Choose **Close**.

Adding a Kinesis Data Firehose Event Destination

The following procedure shows how to add a Kinesis Data Firehose event destination to the configuration set you created.

**To add a Kinesis Data Firehose event destination to the configuration set**

1. Choose the configuration set from the configuration set list.
2. For **Add Destination**, choose **Select a destination type**, and then choose **Kinesis Data Firehose**.
3. For **Name**, type a name for the event destination.
4. Select all **Event types**.
5. Select **Enabled**.
6. For **Stream**, choose the delivery stream that you created in Step 4: Create a Kinesis Data Firehose Delivery Stream (p. 347).
7. For **IAM role**, choose **Let SES make a new role**, and then type a name for the role.
8. Choose **Save**.
9. To exit the **Edit Configuration Set** page, use the back button of your browser.

Next Step

Step 6: Send Emails (p. 350)

Step 6: Send Emails

For Amazon SES to publish events associated with an email, you must specify a configuration set when you send the email. You can also include message tags to categorize the email. This section shows
how to send a simple email that specifies a configuration set and message tags using the Amazon SES console. You send the email to the Amazon SES mailbox simulator so that you can test bounces, complaints, and other email sending outcomes.

**To send an email using the Amazon SES console**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane of the Amazon SES console, under Identity Management, choose Email Addresses.
3. In the list of identities, select the check box of an email address that you have successfully verified with Amazon SES. 
4. Choose Send a Test Email.
5. In the Send Test Email dialog box, for Email Format, choose Raw.
6. For the To address, type an address from the Amazon SES mailbox simulator (p. 181), such as complaint@simulator.amazonses.com or bounce@simulator.amazonses.com.
7. Copy and paste the following message in its entirety into the Message text box, replacing CONFIGURATION-SET-NAME with the name of the configuration set you created in Step 5: Set up a Configuration Set (p. 350), and replacing FROM-ADDRESS with the verified address you are sending this email from.

```
X-SES-MESSAGE-TAGS: campaign=book
X-SES-CONFIGURATION-SET: CONFIGURATION-SET-NAME
Subject: Amazon SES Event Publishing Test
From: Amazon SES User <FROM-ADDRESS>
MIME-Version: 1.0
Content-Type: text/plain

This is a test message.
```
8. Choose Send Test Email.
9. Repeat this procedure a few times so that you generate multiple email sending events. For a few of the emails, change the value of the campaign message tag to clothing to simulate sending for a different email campaign. That way, when you query your Amazon Redshift database for email sending event records in the last step of this tutorial, you can experiment with querying based on email campaign.

**Next Step**

**Step 7: Query Email Sending Events (p. 351)**

**Step 7: Query Email Sending Events**

Now that you have generated some email sending events by sending emails with your configuration set and message tags, you can query those records in Amazon Redshift.

**Note**

We assume that SQL Workbench/J is currently open on your computer, and it is connected to your Amazon Redshift cluster, as described in Step 2: Connect to Your Amazon Redshift Cluster (p. 342).

**To query email sending event data in Amazon Redshift from SQL Workbench/J**

1. To display all of your email sending records, copy the following query and paste it into the Statement 1 window.
select * from ses;

2. Place the cursor within the statement (somewhere before the semicolon), and then choose the Execute current statement button.

You will see the email sending records for all of the emails you sent in Step 6: Send Emails (p. 350). The records in the following figure show that our book campaign had two complaints, and the clothing campaign had one bounce.

3. To count the complaint records for the campaign of type book, copy the following query and paste it into the Statement 1 window.

```
select count(*) as numberOfComplaint from ses where event_type = 'Complaint' and campaign like '%book%';
```

4. Place the cursor within the statement (somewhere before the semicolon), and then choose the Execute current statement button.

The results are the following, showing that the book campaign had two complaints.

---

**Graph email sending events in Amazon CloudWatch**

In this tutorial, you publish Amazon SES email sending events to Amazon CloudWatch and then graph the events using the CloudWatch console.

The following sections walk you through the process.

- Prerequisites (p. 353)
- Step 1: Set up a Configuration Set (p. 353)
• Step 2: Send Emails (p. 354)
• Step 3: Graph Events (p. 355)

Prerequisites

For this tutorial, you will need the following:

• An AWS account – To access any web service that AWS offers, you must first create an AWS account at https://aws.amazon.com/.

• Verified email address – To send emails using Amazon SES, you must verify your “From” address or domain to show that you own it. If you are in the sandbox, you also must verify your “To” addresses. You can verify email addresses or entire domains, but this tutorial requires a verified email address so that you can send an email from the Amazon SES console, which is the simplest way to send an email. For information about how to verify an email address, see Verifying email addresses in Amazon SES (p. 47).

Next Step

Step 1: Set up a Configuration Set (p. 353)

Step 1: Set up a Configuration Set

To set up Amazon SES to publish your email sending events to Amazon CloudWatch, you first create a configuration set, and then you add a CloudWatch event destination to the configuration set. This section shows how to accomplish those tasks.

If you already have a configuration set, you can add a CloudWatch destination to your existing configuration set. In this case, skip to Adding a CloudWatch Event Destination (p. 353).

Creating a Configuration Set

The following procedure shows how to create a configuration set.

To create a configuration set

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, choose Configuration Sets.
3. Choose Create Configuration Set.
4. Type a name for the configuration set, and then choose Create Configuration Set.

Adding a CloudWatch Event Destination

The following procedure shows how to add a CloudWatch event destination to the configuration set you created.

To add a CloudWatch event destination to a configuration set

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane, choose Configuration Sets.
3. Choose the configuration set you created in the previous section.
4. For Add Destination, choose Select a destination type, and then choose CloudWatch.
5. For **Name**, enter a name for the event destination.

6. For **Event types**, choose the metrics that you want to report in Amazon CloudWatch.

7. Choose **Enabled**.

8. For **Value Source**, choose the value that you want to use to categorize the metrics in CloudWatch. For example, if you choose **Message Tag**, you have to specify a key-value pair. Amazon SES sends the selected metrics to CloudWatch if the email contains this key-value pair as a message tag. When you view the metrics in CloudWatch, they’re categorized by the key of the message tag.

   **Note**
   If you choose **Link Tag** as the value source, you can only send click events to CloudWatch. You can use the **Link Tag** value source to determine which links in your emails are clicked most often.

9. Choose **Save**.

10. To exit the **Edit Configuration Set** page, use the back button of your browser.

### Step 2: Send Emails

For Amazon SES to publish events associated with an email, you must specify a configuration set when you send the email. You can also include message tags to categorize the email. This section shows how to send a simple email that specifies a configuration set and message tags using the Amazon SES console. You send the email to the Amazon SES mailbox simulator so that you can test bounces, complaints, and other email sending outcomes.

**To send an email using the Amazon SES console**

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.

2. In the **Navigation** pane of the Amazon SES console, under **Identity Management**, choose **Email Addresses**.

3. In the list of identities, select the check box of an email address that you have successfully verified with Amazon SES (p. 47).

4. Choose **Send a Test Email**.

5. In the **Send Test Email** dialog box, for **Email Format**, choose **Raw**.

6. For the **To** address, type an address from the Amazon SES mailbox simulator (p. 181), such as complaint@simulator.amazonses.com or bounce@simulator.amazonses.com.

7. Copy and paste the following message in its entirety into the **Message** text box, replacing **CONFIGURATION-SET-NAME** with the name of the configuration set you created in Step 1: Set up a Configuration Set (p. 353), and replacing **FROM-ADDRESS** with the verified address you are sending this email from.

```
X-SES-MESSAGE-TAGS: campaign=book
X-SES-CONFIGURATION-SET: CONFIGURATION-SET-NAME
Subject: Amazon SES Event Publishing Test
From: Amazon SES User <FROM-ADDRESS>
MIME-Version: 1.0
Content-Type: text/plain

This is a test message.
```

8. Choose **Send Test Email**.

9. Repeat this procedure a few times so that you generate multiple email sending events. For a few of the emails, change the value of the **campaign** message tag to **clothing** to simulate sending for a different email campaign.
Next Step

Step 3: Graph Email Sending Events (p. 355)

Step 3: Graph Email Sending Events

Now that you have published some Amazon SES email sending events to CloudWatch by sending emails with your configuration set and message tags, you can graph metrics for those events using the CloudWatch console.

To graph email sending event metrics

1. Sign in to the AWS Management Console and open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. In the left navigation pane, choose Metrics.
3. In the All metrics tab, choose SES.
   Tip
   You can also type SES into the search field.
4. Choose the value source that you specified in Adding a CloudWatch Event Destination (p. 353). For example, if you specified the message tag "category:books" as the value source, choose category.
5. Choose the metric that you want to view. A graph appears in the details pane.

Analyze email sending events with Amazon Kinesis Data Analytics

Amazon Kinesis Data Analytics enables you to process and analyze streaming data using SQL. You can use Amazon Kinesis Data Analytics to analyze your Amazon SES email sending events.

In this tutorial, you first set up an Amazon SES configuration set to publish your email sending events to an Amazon Kinesis Data Firehose delivery stream, and then you send emails through Amazon SES using that configuration set. You then set up Amazon Kinesis Data Analytics to capture the email sending events from the Kinesis Data Firehose stream and use SQL to extract key information from the emails you sent.

Note
This tutorial requires that you have an application that can send a steady stream of emails through Amazon SES. This requirement is explained in Prerequisites (p. 355).

The following sections walk you through the tutorial.

- Prerequisites (p. 355)
- Step 1: Create a Kinesis Data Firehose Delivery Stream (p. 356)
- Step 2: Set up a Configuration Set (p. 357)
- Step 3: Send Emails (p. 358)
- Step 4: Create an Amazon Kinesis Data Analytics Application (p. 359)
- Step 5: Run a SQL Query (p. 363)
- (Optional) Step 6: Save SQL Query Results (p. 364)

Prerequisites

For this tutorial, you need the following:
• **An AWS account** – To access any web service that AWS offers, you must first create an AWS account at [https://aws.amazon.com/](https://aws.amazon.com/).

• **Verified email address** – To send emails using Amazon SES, you must verify your "From" address or domain to show that you own it. If you are in the sandbox, you also must verify your "To" addresses. You can verify email addresses or entire domains, but this tutorial requires a verified email address so that you can send an email from the Amazon SES console, which is the simplest way to send an email. For information about how to verify an email address, see Verifying email addresses in Amazon SES (p. 47).

• **Email application** – To use Amazon Kinesis Data Analytics as described in this tutorial, you must send a steady stream of emails through Amazon SES so that you generate a steady stream of email sending events. This enables Amazon Kinesis Data Analytics to automatically detect the schema and then to process the event records with SQL. Sending one email every ten seconds for five minutes is sufficient for this tutorial.

  **Important**
  If you do not have an existing email campaign to send to real recipients, we strongly recommend that you send emails to an Amazon SES mailbox simulator (p. 181) address. Emails that you send to the mailbox simulator do not count toward your Amazon SES bounce and complaint rates or your daily sending quota.

**Next Step**

**Step 1: Create a Kinesis Data Firehose Delivery Stream** (p. 356)

**Step 1: Create a Kinesis Data Firehose Delivery Stream**

To analyze Amazon SES email sending events with Amazon Kinesis Data Analytics, you must configure Amazon SES to publish the events to an Amazon Kinesis Data Firehose delivery stream, and then configure Amazon Kinesis Data Analytics to get the event data from Kinesis Data Firehose.

When you set up a Kinesis Data Firehose delivery stream, you choose the final destination of the data. Your destination options are Amazon Simple Storage Service (Amazon S3), Amazon OpenSearch Service, and Amazon Redshift. If you simply want to analyze email sending events with Amazon Kinesis Data Analytics, it does not matter which destination you choose. For this tutorial, we configure Kinesis Data Firehose to publish the data to Amazon S3, but you can use the other destination options if they are in the same region as your Amazon SES sending and Kinesis Data Firehose delivery stream.

This section shows how to create a Kinesis Data Firehose delivery stream using the Kinesis Data Firehose console. For this tutorial, we choose basic options. For information about all available options, see Creating an Amazon Kinesis Firehose Delivery Stream in the [Amazon Kinesis Data Firehose Developer Guide](https://docs.aws.amazon.com/kinesis消防 hose/latest/dev/).  

**To create a delivery stream from Kinesis Data Firehose to Amazon S3**

1. Sign in to the AWS Management Console and open the Kinesis Data Firehose console at [https://console.aws.amazon.com/firehose/](https://console.aws.amazon.com/firehose/).
2. Choose Create Delivery Stream.
3. On the Destination page, choose the following options.
   
   • **Destination** – Choose Amazon S3.
   
   • **Delivery stream name** – Type a name for the delivery stream.
   
   • **S3 bucket** – Choose an existing bucket, or choose New S3 Bucket. If you create a new bucket, type a name for the bucket and choose the region your console is currently using.
   
   • **S3 prefix** – Leave this field empty.
4. Choose Next.
5. On the **Configuration** page, leave the fields at the default settings. The only required step is to select an IAM role that enables Kinesis Data Firehose to access your resources, as follows:

   a. For **IAM Role**, choose **Select an IAM role**.
   
   b. In the drop-down menu, under **Create/Update existing IAM role**, choose **Firehose delivery IAM role**.

   You are taken to the IAM console.

   c. In the IAM console, leave the fields at their default settings, and then choose **Allow**.

You return to the Kinesis Data Firehose delivery stream set-up steps in the Kinesis Data Firehose console.

6. Choose **Next**.

7. On the **Review** page, review your settings, and then choose **Create Delivery Stream**.

---

**Next Step**

**Step 2: Set up a Configuration Set (p. 357)**

**Step 2: Set up a Configuration Set**

To set up Amazon SES to publish your email sending events to Amazon Kinesis Data Firehose, you create a configuration set, and then you add a Kinesis Data Firehose event destination to the configuration set. This section describes how to accomplish those tasks.

If you already have a configuration set, you can add a Kinesis Data Firehose destination to your existing configuration set. In this case, skip to **Adding a Kinesis Data Firehose Event Destination (p. 358)**.

**Creating a Configuration Set**

The following procedure describes how to create a configuration set.
To create a configuration set

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the left navigation pane, choose Configuration Sets.
3. In the content pane, choose Create Configuration Set.
4. Type a name for the configuration set, and then choose Create Configuration Set.
5. Choose Close.

Adding a Kinesis Data Firehose Event Destination

The following procedure shows how to add a Kinesis Data Firehose event destination to the configuration set you created.

To add a Kinesis Data Firehose event destination to the configuration set

1. Choose the configuration set from the configuration set list.
2. For Add Destination, choose Select a destination type, and then choose Kinesis Data Firehose.
3. For Name, type a name for the event destination.
4. Select all Event types.
5. Select Enabled.
6. For Stream, choose the delivery stream that you created in Step 1: Create a Kinesis Data Firehose Delivery Stream (p. 356).
7. For IAM role, choose Let SES make a new role, and then type a name for the role.
8. Choose Save.
9. To exit the Edit Configuration Set page, use the back button of your browser.

Next Step

Step 3: Send Emails (p. 358)

Step 3: Send Emails

Because this tutorial uses the Amazon Kinesis Data Analytics console to process and analyze streaming data, you must set up a steady stream of emails through Amazon SES. This tutorial assumes that you have an application that can send these emails. Sending one email every ten seconds for five minutes is sufficient for this tutorial. We highly recommend that you use a "To" address from the Amazon SES mailbox simulator (p. 181), such as success@simulator.amazonses.com.

To enable event publishing for an email, you provide the name of the configuration set to Amazon SES when you send the email. You can optionally include message tags to categorize the email. You provide this information to Amazon SES as either parameters to the email sending API, Amazon SES-specific email headers, or custom headers in your MIME message. For more information, see Send Email Using Amazon SES Event Publishing (p. 298).

For example, you might add the following Amazon SES-specific email headers to your email to simulate a book campaign. Replace CONFIGURATION-SET-NAME with the name of the configuration set you created in Step 2: Set up a Configuration Set (p. 357).

```
X-SES-CONFIGURATION-SET: CONFIGURATION-SET-NAME
X-SES-MESSAGE-TAGS: campaign=book
```
Next Step

Step 4: Create an Amazon Kinesis Data Analytics Application (p. 359)

Step 4: Create an Amazon Kinesis Data Analytics Application

Now that you have set up event publishing with Amazon SES, you can configure Amazon Kinesis Data Analytics to capture the email sending event data from your Amazon Kinesis Data Firehose delivery stream. To do this, you create an Amazon Kinesis Data Analytics application.

The following procedure shows how to use the Amazon Kinesis Data Analytics console to create an application that captures Amazon SES email sending event data from your Kinesis Data Firehose delivery stream, and then how to perform a simply SQL query on the data to return the events of type "Send".

**Note**
The email sending events of different event types (send, bounce, complaint, and delivery) have different JSON schemas (p. 301). In a production environment, you might examine several fields of this schema, but in this tutorial, we limit our examination to a small set of fields that are present for all event types.

To create an Amazon Kinesis Data Analytics application

1. Start sending a steady stream of emails configured for event publishing through Amazon SES, and continue sending the emails throughout this procedure. This is required so that Amazon Kinesis Data Analytics can automatically detect the schema of the event records. Sending one email every ten seconds for five minutes is sufficient for this tutorial. For more information, see Step 3: Send Emails (p. 358).

   After your email program has sent a few emails, move to the next step.

2. Sign in to the AWS Management Console and open the Kinesis Data Analytics console at [https://console.aws.amazon.com/kinesisanalytics](https://console.aws.amazon.com/kinesisanalytics).

3. Choose Create new application.

4. Enter an application name and description, and then choose Save and continue.

5. Choose Connect to a source.

6. Choose the Kinesis Data Firehose stream you created in Step 2: Set up a Configuration Set (p. 357).

   Amazon Kinesis Data Analytics attempts to discover the schema of the email sending event records based on the incoming records. If Amazon Kinesis Data Analytics displays Error discovering input schema, that means that Amazon Kinesis Data Analytics has not received any email sending records yet. Choose Rediscover schema. You might need to choose this button several times. If schema discovery does not succeed after several attempts, ensure that your email sending application is steadily sending emails, and that the emails specify a configuration set.

    When Amazon Kinesis Data Analytics detects a schema, it displays a success message and lists the records it detected.

    **Important**

    Do not choose Save and continue. This will cause errors because the discovered schema does not adhere to SQL naming constraints. You must edit the schema as described in the next step.

7. Choose Edit schema.
8. For this tutorial, we remove most of the rows. Choose X next to all rows except rows with the following column names:

- eventType
- timestamp
- messageId
- to
- ses:configuration-set

**Important**
Do not choose Save schema and update stream samples. This will cause errors because the discovered schema does not adhere to SQL naming constraints. You must edit the schema as described in the next step.
9. Examine the remaining entries under **Column name** and compare them to the SQL naming requirements as follows:

- **Format** – As described in Identifiers in the *Amazon Kinesis Data Analytics SQL Reference*, unquoted identifiers must start with a letter or underscore, and be followed by letters, digits, or underscores. Amazon SES auto-tag names do not comply with these requirements because they contain colons and dashes. You will edit these in the next step.

- **Reserved words** – Column names must not conflict with the SQL reserved words listed in Reserved Words and Keywords in the *Amazon Kinesis Data Analytics SQL Reference*. Examples of reserved keywords that conflict with Amazon SES event records are `timestamp`, `value`, `date`, `from`, and `to`.

10. Edit the remaining column names to conform to the SQL requirements as follows:

    - Rename `ses:configuration-set` to `ses_configuration_set`.
    - Rename `timestamp` to `ses_timestamp`.
    - Rename `to` to `ses_to`.

![Kinesis Analytics dashboard](image)
11. Choose **Save schema and update stream samples**. If you encounter validation errors, ensure that you correctly performed step 10. If you encounter the **No rows in source stream** error, ensure that you are still sending the email stream that you started at the beginning of this procedure, and then choose **Retrieve rows**. You might need to choose **Retrieve rows** several times before Amazon Kinesis Data Analytics captures records.

12. Upon successful retrieval of rows, choose **Exit (done)**.

**Next Step**

**Step 5: Run a SQL Query (p. 363)**
Step 5: Run a SQL Query

Now that you have created an Amazon Kinesis Data Analytics application and configured it to use your Amazon Kinesis Data Firehose delivery stream as its source, you can query the email sending event data that the Kinesis Data Firehose delivery stream receives.

This topic shows how to run a SQL query on the email sending event data.

**Important**

This procedure requires that you continue to send a steady stream of emails configured for event publishing through Amazon SES, as described in Step 3: Send Emails (p. 358).

To run a SQL query in Amazon Kinesis Data Analytics

1. Assuming that you have moved on to this procedure after completing the last step (p. 359), go to the Amazon Kinesis Data Analytics console top menu and choose your application.

2. Choose **Go to SQL editor**.

Amazon Kinesis Data Analytics attempts to read event data from the Kinesis Data Firehose stream. If you encounter the **No rows in source stream** error, ensure that you are still sending the email stream you started at the beginning of this procedure, and then choose **Retrieve rows**.

3. In the code editor box, paste the following.
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" ("eventType" VARCHAR(16), "ses_timestamp" timestamp, "messageId" VARCHAR(64), "ses_to" VARCHAR(64), "ses_configuration_set" VARCHAR(32));

CREATE OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
SELECT STREAM "eventType", "ses_timestamp", "messageId", "ses_to", "ses_configuration_set"
FROM "SOURCE_SQL_STREAM_001"
WHERE "eventType" = 'Send'

4. Choose **Save and run SQL**.

After Amazon Kinesis Data Analytics retrieves and processes incoming records, you see a list of event records of type "Send".

Next Step

(Optional) Step 6: Save SQL Query Results (p. 364)

(Optional) Step 6: Save SQL Query Results

You can set up your Amazon Kinesis Data Analytics application to write the output of your SQL queries to an Amazon Kinesis Data Firehose delivery stream. To do so, you must create another Kinesis Data Firehose delivery stream because you cannot use the same delivery stream as both the source and destination of an Amazon Kinesis Data Analytics application. As with any Kinesis Data Firehose delivery stream, you can choose Amazon Simple Storage Service (Amazon S3), Amazon OpenSearch Service, or Amazon Redshift as the destination.

The following procedure shows how to configure Amazon Kinesis Data Analytics to save SQL query results in JSON format to a Kinesis Data Firehose delivery stream that writes the data to Amazon S3. Then you run a SQL query and access the saved data.
To save the results of SQL queries to Amazon S3

1. Set up a new Kinesis Data Firehose stream that uses Amazon S3 as the destination. It is the same procedure as Step 1: Create a Kinesis Data Firehose Delivery Stream (p. 356).

2. Go to the Amazon Kinesis Data Analytics console, choose the arrow next to your application, and then choose Application details.

3. Choose Connect to a destination.

4. Choose the Kinesis Data Firehose stream you created in step 1, leave the rest of the options at their default settings, and then choose Save and continue.

In several seconds, you return to the main page of the application.
5. Choose **Go to SQL results**.

6. Choose **Save and run SQL** to re-run the query you ran in *Step 5: Run a SQL Query* (p. 363).

Amazon Kinesis Data Analytics attempts to process event data it receives from the Kinesis Data Firehose delivery stream. If you encounter the **No rows have arrived yet** error, ensure that you are still sending emails so that Amazon Kinesis Data Analytics has email sending events to process.

As Amazon Kinesis Data Analytics processes records, results appear in the **Real-time analytics** tab. Amazon Kinesis Data Analytics automatically saves the results to the Amazon S3 bucket that you specified when you set up the Kinesis Data Firehose delivery stream in step 1.
7. To retrieve the results, go to the Amazon S3 console.
8. Choose the Amazon S3 bucket that is associated with the Kinesis Data Firehose delivery stream that the Amazon Kinesis Data Analytics application uses as its destination.
9. Navigate to the data, which, by default, is organized in a folder hierarchy based on the date the results are saved to the bucket.

If the bucket is empty, wait a few minutes and try again. It can take several minutes for data to get from Amazon Kinesis Data Analytics to your Amazon S3 bucket.
10. Choose a file, and then from the Actions menu, choose Download.

11. Follow the on-screen instructions to download the file to your computer.
12. On your computer, open the file with a text editor. The records are in JSON format, and each record is contained in curly braces. The following is an example of a file that contains two records.
Monitoring your Amazon SES sender reputation

Amazon SES actively tracks several metrics that may cause your reputation as a sender to be damaged, or that could cause your email delivery rates to decline. Two important metrics that we consider in this process are the bounce and complaint rates for your account. If the bounce or complaint rates for your account are too high, we might place your account under review or pause your account's ability to send email.

Because your bounce and complaint rate are so important to the health of your account, Amazon SES includes a reputation dashboard that you can use to track these metrics. The reputation dashboard can also display information about factors unrelated to bounces or complaints that could damage your sender reputation. For example, if you send email to a known spamtrap, you will see a message on this dashboard.

This section contains information about accessing the reputation dashboard, interpreting the information it contains, and setting up systems to actively notify you of factors that could impact your sender reputation.

In this section, you will find the following topics:
- Using the reputation dashboard to track bounce and complaint rates (p. 369)
- Reputation dashboard messages (p. 370)
- Creating reputation monitoring alarms using CloudWatch (p. 382)
- SNDS metrics for dedicated IPs (p. 385)
- Automatically pausing email sending (p. 386)

Using the reputation dashboard to track bounce and complaint rates

The reputation dashboard contains the same information that the Amazon SES team sees when determining the health of individual accounts.

To view the reputation dashboard

1. Sign in to the AWS Management Console and open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. In the navigation pane on the left side of the screen, choose Reputation Dashboard.

The dashboard displays the following information:

- **Account status** – A brief description of the health of your account. Possible values include:
  - **Healthy** – There are no issues currently impacting your account.
  - **Under review** – Your account is under review. If the issues that caused us to place your account under review aren’t resolved by the end of the review period, we might pause your account's ability to send email.
  - **Pending end of review decision** – Your account is under review. Because of the nature of the issues that caused us to place your account under review, we need to perform a manual review of your account before we take any further action.
• **Sending paused** – We’ve paused your account’s ability to send email. While your account’s ability to send email is paused, you won’t be able to send email using Amazon SES. You can request that we review this decision. To learn more about requesting a review, see Amazon SES Sending review process FAQs (p. 489).

• **Pending sending pause** – Your account is under review. The issues that caused us to place your account under review haven’t been resolved. In this situation, we typically pause your account’s ability to send email. However, because of the nature of your account, we need to review your account before any further action is taken.

• **Bounce Rate** – The percentage of emails sent from your account that resulted in a hard bounce.

• **Complaint Rate** – The percentage of emails sent from your account that resulted in recipients reporting them as spam.

**Note**
The **Bounce Rate** and **Complaint Rate** sections also include status messages for their respective metrics. The following is a list of status messages that may be displayed for these metrics:

• **Healthy** – The metric is within normal levels.

• **Almost healed** – The metric caused your account to be placed under review. Since the review period began, the metric has stayed below the maximum rate. If the metric remains below the maximum rate, the status of this metric changes to **Healthy** before the review period ends.

• **Under review** – The metric caused your account to be placed under review, and is still above the maximum rate. If the issue that caused the metric to exceed the maximum rate is not resolved by the end of the review period, we might pause your account’s ability to send email.

• **Sending pause** – The metric caused us to pause your account’s ability to send email. While your account’s ability to send email is paused, you can’t send email using Amazon SES. You can request that we review this decision. To learn more about submitting a request for review, see Amazon SES Sending review process FAQs (p. 489).

• **Pending sending pause** – The metric caused us to place your account under review. The issues that caused this review period haven’t been resolved. These issues might cause us to pause your account’s ability to send email. A member of the Amazon SES team has to review your account before we take any further action.

• **Other Notifications** – If your account is experiencing reputation-related issues that are not related to bounces or complaints, a brief message will be shown here. For more information about the notifications that can be shown in this area, see Reputation dashboard messages (p. 370).

**Note**
The reputation dashboard is available to all users who have access to the AWS console. You can’t use IAM policies to restrict access to the reputation dashboard.

---

**Reputation dashboard messages**

The Amazon SES reputation dashboard provides important metrics related to your account. The following sections describe the messages that might be displayed in this dashboard, and provide tips and information that you might be able to use to resolve issues related to your sender reputation.

This section contains information about the following types of notifications:

• **Status Messages** (p. 371)

• **Bounce Rate Notification** (p. 372)

• **Complaint Rate Notification** (p. 373)
• Anti-Spam Organization Notification (p. 374)
• Direct Feedback Notification (p. 374)
• Domain Blocklist Notification (p. 375)
• Internal Review Notification (p. 376)
• Mailbox Provider Notification (p. 377)
• Recipient Feedback Notification (p. 378)
• Related Account Notification (p. 379)
• Spamtrap Notification (p. 380)
• Vulnerable Site Notification (p. 381)
• Other Notification (p. 381)

Status Messages

When you use the reputation dashboard, you see a message describing the status of your Amazon SES account. The following is a list of possible account status values:

• **Healthy** – There are no issues currently impacting your account.
• **Under review** – Your account is under review. If the issues that caused us to place your account under review aren't resolved by the end of the review period, we might pause your account’s ability to send email.
• **Pending end of review decision** – Your account is under review. Because of the nature of the issues that caused us to place your account under review, we need to perform a manual review of your account before we take any further action.
• **Sending paused** – We’ve paused your account’s ability to send email. While your account’s ability to send email is paused, you won’t be able to send email using Amazon SES. You can request that we review this decision. To learn more about requesting a review, see Amazon SES Sending review process FAQs (p. 489).
• **Pending sending pause** – The issues that caused us to place your account under review haven’t been resolved. In this situation, we typically pause your account’s ability to send email. However, because of the nature of your account, we need to review your account before any further action is taken.

Additionally, the **Bounce Rate** and **Complaint Rate** sections of the reputation dashboard display status summaries for their respective metrics. The following is a list of possible metric status values:

• **Healthy** – The metric is within normal levels.
• **Almost healed** – The metric caused your account to be placed under review. Since the review period began, the metric has stayed below the maximum rate. If the metric remains below the maximum rate, the status of this metric changes to **Healthy** before the review period ends.
• **Under review** – The metric caused your account to be placed under review, and is still above the maximum rate. If the issue that caused the metric to exceed the maximum rate is not resolved by the end of the review period, we might pause your account’s ability to send email.
• **Sending pause** – The metric caused us to pause your account's ability to send email. While your account's ability to send email is paused, you can't send email using Amazon SES. You can request that we review this decision. To learn more about submitting a request for review, see Amazon SES Sending review process FAQs (p. 489).
• **Pending sending pause** – The metric caused us to place your account under review. The issues that caused this review period haven't been resolved. These issues might cause us to pause your account's ability to send email. A member of the Amazon SES team has to review your account before we take any further action.
Bounce Rate Notification

This section contains additional information about bounce rate notifications shown in the Amazon SES reputation dashboard.

Why you received this notification

You received this notification because the bounce rate for your account was too high. The bounce rate is based on the number of hard bounces generated by your Amazon SES account. Email providers interpret a high bounce rate as a sign that a sender isn't properly managing their recipient list, and that the sender might be sending unsolicited email.

A hard bounce occurs when an email is sent to an address that doesn't exist. Amazon SES doesn't consider soft bounces (which occur when a recipient's address is temporarily unable to receive messages) in this calculation. Bounced emails that you send to verified addresses and domains, as well as emails that you send to the Amazon SES inbox simulator (p. 181), also aren't considered in this calculation.

We calculate your bounce rate based on a representative volume of email. A representative volume is an amount of email that represents your typical sending practices. To be fair to both high- and low-volume senders, the representative volume is different for each account and changes as the account's sending patterns change.

For best results, maintain a bounce rate below 5%. Higher bounce rates can impact the delivery of your emails. If your bounce rate is 5% or greater, we automatically place your account under review. If your bounce rate is 10% or greater, we might pause your account's ability to send additional email until you resolve the issue that caused the high bounce rate.

What you can do to resolve the issue

If you haven't done so already, put a process in place to capture and manage bounces and complaints. All Amazon SES accounts are required to have these processes in place. For more information, see Email program success metrics (p. 465).

Next, determine which email addresses are bouncing, and create and implement a plan for reducing or eliminating these bounces. If your account's ability to send email has already been paused, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf.

If your account is under review

At the end of the review period, if the bounce rate for your account remains above 10%, we might pause your account's ability to send email until you resolve the issue.

If you have implemented changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. In your response to the case, describe the changes you implemented. If we agree that the changes will reduce your bounce rate, we adjust our calculations to only consider bounces received after your changes were implemented.

If your account's ability to send email is paused

You can request that we reconsider this decision. For more information, see Amazon SES Sending review process FAQs (p. 489).

When you implement changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. Include details of the actions you have taken to resolve this issue, as well as details of your plans to ensure that this issue doesn't occur again. After we receive your request, we review the information that you provided and change the status of your account if necessary.
Complaint Rate Notification

This section contains additional information about complaint rate notifications shown in the Amazon SES reputation dashboard.

Why you received this notification

You received this notification because the complaint rate for your account was too high. The complaint rate is based on the number of complaints generated by your Amazon SES account. Email providers interpret a high complaint rate as a sign that a sender isn't properly managing their recipient list, and that the sender might be sending unsolicited email.

A complaint occurs when a recipient identifies an email that you sent as spam. This usually occurs when the recipient uses the Report Spam button in their email client. Complaints that are generated by emails that you send to the Amazon SES inbox simulator (p. 181) aren't considered in this calculation.

We calculate your complaint rate based on a representative volume of email. A representative volume is an amount of email that represents your typical sending practices. To be fair to both high- and low-volume senders, the representative volume is different for each account and changes as the account's sending patterns change.

For best results, maintain a complaint rate below 0.1%. Higher complaint rates can impact the delivery of your emails. If your complaint rate is 0.1% or greater, we automatically place your account under review. If your complaint rate is 0.5% or greater, we might pause your account's ability to send additional email until you resolve the issue that caused the high complaint rate.

What you can do to resolve the issue

If you haven't done so already, put a process in place to capture and manage bounces and complaints. All Amazon SES accounts are required to have these processes in place. For more information, see Email program success metrics (p. 465).

Next, determine which messages you are sending that result in complaints, and implement a plan for reducing these complaints. If your account's ability to send email has already been paused, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf.

While you should immediately stop sending to addresses that have complained, it is important that you identify the factors that are causing recipients to issue complaints. After you identify these factors, adjust your email sending behaviors to address them.

If your account is under review

At the end of the review period, if the complaint rate for your account remains above 0.5%, we might pause your account's ability to send email until you resolve the issue.

If you have implemented changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. In your response to the case, describe the changes you implemented. If we agree that the changes will reduce your complaint rate, we adjust our calculations to only consider the complaints that were received after you implemented the changes.

If your account's ability to send email is paused

You can request that we reconsider this decision. For more information, see Amazon SES Sending review process FAQs (p. 489).

When you have implemented changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. Include details of the actions you
have taken to resolve this issue, as well as details of your plans to ensure that this issue doesn't occur again. After we receive your request, we review the information that you provided and change the status of your account if necessary.

**Anti-Spam Organization Notification**

This section contains additional information about anti-spam organization notifications shown in the Amazon SES reputation dashboard.

**Why you received this notification**

A reputable anti-spam organization has reported that some of the content being sent from your Amazon SES account has been flagged as unsolicited or problematic by their systems.

We're unable to provide information about the specific messages that caused the anti-spam organization to flag your content as problematic. We can't provide the name of the organization that issued the report. Typically, anti-spam organizations consider a combination of the following factors: recipient feedback, message engagement metrics, attempted deliveries to invalid addresses, content that is flagged by their spam filters, and spamtrap hits. This isn't an exhaustive list; other factors might cause these organizations to flag your content.

**What you can do to resolve the issue**

To resolve this issue, you need to determine what aspects of your email sending program might be causing the anti-spam organization to flag your email as problematic. You then need to change your sending program to address those issues.

**If your account is under review**

At the end of the review period, if the anti-spam organization continues to identify the email sent from your account as problematic, we might pause your account's ability to send email until you resolve the issue.

If you have implemented changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. In your message, provide details of the changes you made. When we receive this information, we will extend the review period to ensure that we're only analyzing the anti-spam organization notifications we have received after you implemented your changes. At the end of this extended review period, your account is no longer listed by the anti-spam organization, we will remove the review period for your account.

**If your account's ability to send email is paused**

You can request that we reconsider this decision. For more information, see Amazon SES Sending review process FAQs (p. 489).

When you have implemented changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. Include details of the actions you have taken to resolve this issue, as well as details of your plans to ensure that this issue doesn't occur again. After we receive your request, we review the information that you provided and change the status of your account if necessary.

**Direct Feedback Notification**

This section contains additional information about direct feedback notifications shown in the Amazon SES reputation dashboard.
Why you received this notification

A significant number of users have contacted Amazon SES directly to report messages that they received from an address or domain associated with your Amazon SES account. This type of feedback isn't visible in the complaints reported by mailbox providers directly, and isn't included in the bounce and complaint metrics shown on the reputation dashboard.

To protect the privacy of the users who reported these issues, we can't provide their email addresses.

Recipients can complain to Amazon SES when they receive messages that they didn't sign up to receive, when they don't receive the type of mail they expected to receive, when they don't find the email they receive to be useful or interesting, when they don't recognize that the messages are something that they signed up for, or when they are receiving too many messages. This list isn't exhaustive; the factors that are relevant in your case depend on your specific email sending program.

What you can do to resolve the issue

We recommend that you implement a double opt-in strategy, as described in Building and maintaining your lists (p. 468), for acquiring new addresses, and that you only send email to addresses that complete the double opt-in process.

Additionally, you should purge your lists of addresses that haven't interacted with your emails recently. You can use open and click tracking, as described in Monitoring your Amazon SES sending activity (p. 259), to determine which users are viewing and interacting with the content you send.

If your account is under review

At the end of the review period, if Amazon SES continues to receive a significant number of direct complaints about messages sent from your account, we might pause your account's ability to send email until you resolve the issue.

If you have implemented changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. Provide detailed information about the steps you've taken to resolve the issue, and describe how these steps prevent the issue from happening again in the future. If we agree that the changes you've made appropriately address the issue, we cancel the review period on your account.

If your account's ability to send email is paused

You can request that we reconsider this decision. For more information, see Amazon SES Sending review process FAQs (p. 489).

When you have implemented changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. Include details of the actions you have taken to resolve this issue, as well as details of your plans to ensure that this issue doesn't occur again. After we receive your request, we review the information that you provided and change the status of your account if necessary.

Domain Blocklist Notification

This section contains additional information about domain blocklist notifications shown in the Amazon SES reputation dashboard.

Why you received this notification

Emails sent from your Amazon SES account contain references to domains that have been listed on a reputable Domain Blocklist. Domains on these lists are typically associated with abusive or malicious
behavior. The domains in question might or might not be the domains from which you are sending email. Messages that include references or links to a domain on a blocklist, or that include images hosted on such a domain, might also be flagged.

We're unable to provide the names of the domains that are causing your messages to be flagged, or to identify which emails were flagged in this way.

**What you can do to resolve the issue**

First, create a list of all of the domains referenced in the emails you send through Amazon SES. Next, use the Spamhaus Domain Lookup tool to determine which domains in your email are on the domain blocklist. More than one domain referenced in the emails you send might be on this blocklist.

The Spamhaus Domain Blocklist isn't affiliated with Amazon SES or AWS. We make no guarantees about the accuracy of the domains on this list. The Spamhaus Domain Blocklist and Domain Lookup Tool are owned, operated, and maintained by the Spamhaus Project.

**If your account is under review**

We look for references to domains that have been used for malicious purposes in the emails that you send during the review period. If your emails still contain a significant number of references to these domains, we might pause your account's ability to send email until you resolve the issue.

If you have implemented changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. In your message, provide details of the changes you made. When we receive this information, we extend the review period to ensure that we're only analyzing the number of blocklisted domains present in your email after you put your changes in place. At the end of this extended review period, if the number of domain blocklist notifications has been reduced or eliminated, and we believe that you've taken steps to prevent this issue from occurring again in the future, we cancel the review period for your account.

**If your account's ability to send email is paused**

You can request that we reconsider this decision. For more information, see Amazon SES Sending review process FAQs (p. 489).

When you have implemented changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. Include details of the actions you have taken to resolve this issue, as well as details of your plans to ensure that this issue doesn't occur again. After we receive your request, we review the information that you provided and change the status of your account if necessary.

**Internal Review Notification**

This section contains additional information about internal review notifications shown in the Amazon SES reputation dashboard.

**Why you received this notification**

A comprehensive review of your account identified several characteristics that may cause mailbox providers or recipients to identify your messages as spam.

To protect our abuse detection process, we can't reveal the specific factors that led to your account being flagged in this way.

Common factors that can lead to this determination include the following:
• Messages being flagged by commercial anti-spam systems.
• Message content that implies the recipient hasn't explicitly requested the email.
• Mismatches between the message sender and the branding within the email body.
• Content that doesn't make it obvious who the sender is.
• Sending messages that deal with content that is associated with unsolicited email.
• Formatting patterns associated with unsolicited email.
• Sending from or making reference to domains with poor reputations.

This isn't a comprehensive list. The specific reason for this notification might be a combination of any of these factors, or the reason might be something not listed.

What you can do to resolve the issue

The following suggestions might help reduce the severity of the issue:

• Ensure that the only recipients you are contacting are those who have explicitly asked to receive email from you.
• Never purchase, rent, or borrow lists of email recipients.
• Don't attempt to hide your identity or the purpose of your communication in the messages you send.
• Create a list of all of the domains referenced in the emails you send through Amazon SES, and then use the Spamhaus Domain Lookup tool at https://www.spamhaus.org/lookup/ to determine if any of those domains are on the Spamhaus Domain Blocklist.
• Ensure that you are following industry best practices when designing your emails.

This list isn't exhaustive, but it should help you identify some of the most common factors that might lead to your email being flagged.

The Spamhaus Domain Blocklist isn't affiliated with Amazon SES or AWS. We make no guarantees about the accuracy of the domains on this list. The Spamhaus Domain Blocklist and Domain Lookup Tool are owned, operated, and maintained by the Spamhaus Project.

If your account is under review, or if your account's ability to send email is paused

When you have implemented changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. Provide detailed information about the steps you've taken to resolve the issue, and describe how these steps prevent the issue from happening again in the future. If we agree that the changes you've made appropriately address the issue, we cancel the review period or remove the sending pause from your account.

If we remove a review period or sending pause from your account, and we observe the same issue at a later time, we might place your account under review or pause your ability to send email again. In extreme cases, or if we observe repeated instances of the same issue, we might permanently suspend your account's ability to send email.

See Amazon SES Sending review process FAQs (p. 489) for more information about what to do if your account is under review, or your account's ability to send email is paused.

Mailbox Provider Notification

This section contains additional information about mailbox provider notifications shown in the Amazon SES reputation dashboard.
Why you received this notification

A major mailbox provider has reported to us that unsolicited or malicious email is being sent from an address or domain associated with your Amazon SES account.

We can't share the identity of the organization that issued this report. Additionally, we don't have information about the specific factors that caused the mailbox provider to issue the report. Typically, mailbox providers make this kind of determination based on customer feedback, customer engagement metrics, attempted deliveries to invalid addresses, and content that is flagged by spam filters. This list isn't exhaustive; there might be other factors that caused the mailbox provider to flag your content.

What you can do to resolve the issue

To resolve this issue, you need to determine which aspects of your email sending program might have caused mailbox providers to flag your mail as being problematic. You must then change your sending program to address those issues.

If your account is under review

At the end of the review period, if the mailbox provider continues to identify the email sent from your account as being problematic, we might pause your account's ability to send email until you resolve the issue.

If you have implemented changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. In your message, provide details of the changes you made. When we receive this information, we will extend the review period to ensure that we're only analyzing the number of mailbox provider notifications we receive after you implement your changes. At the end of this extended review period, if the mailbox provider no longer reports your account as being problematic, we might remove the review from your account.

If your account's ability to send email is paused

You can request that we reconsider this decision. For more information, see Amazon SES Sending review process FAQs (p. 489).

When you have implemented changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. Include details of the actions you have taken to resolve this issue, as well as details of your plans to ensure that this issue doesn't occur again. After we receive your request, we review the information that you provided and change the status of your account if necessary.

Recipient Feedback Notification

This section contains additional information about recipient feedback notifications shown in the Amazon SES reputation dashboard.

Why you received this notification

A major mailbox provider has reported to us that large numbers of their users are reporting mail sent from your Amazon SES account as unsolicited. This type of feedback isn't visible in the complaints reported by mailbox providers directly, and isn't included in the Amazon SES bounce and complaint notifications.

A large number of complaints can have a negative impact on all Amazon SES users. To protect your reputation and that of other Amazon SES customers, we take immediate action when an account receives a certain number of complaints.
We are unable to provide a list of the specific email addresses that are reporting your email as unsolicited. Additionally, we're unable to share the name of the mailbox provider that has reported this issue to us.

What you can do to resolve the issue

To resolve this issue, you need to determine which aspects of your email sending program might be causing your recipients to issue complaints against the email messages they receive from you. After you identify these factors, change your email sending practices to correct them.

To acquire new addresses, we recommend that you implement a double opt-in strategy, as described in Building and maintaining your lists (p. 468). We recommend that you only send email to addresses that have completed the double opt-in process.

Additionally, you should purge your lists of addresses that haven't interacted with your emails recently. You can use open and click tracking, as described in Monitoring your Amazon SES sending activity (p. 259), to determine which users are viewing and interacting with the content you send.

If your account is under review

At the end of the review period, if the mailbox provider continues to report a significant number of complaints, we might pause your account's ability to send email until you resolve the issue.

If you have implemented changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. In your message, provide details of the changes you made. When we receive this information, we extend the review period to ensure that we're only analyzing the number of mailbox provider complaints that we receive after you implement your changes. At the end of this extended review period, if the number of mailbox provider complaints has been reduced or eliminated, we might remove the review from your account.

If your account's ability to send email is paused

You can request that we reconsider this decision. For more information, see Amazon SES Sending review process FAQs (p. 489).

When you have implemented changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. Include details of the actions you have taken to resolve this issue, as well as details of your plans to ensure that this issue doesn't occur again. After we receive your request, we review the information that you provided and change the status of your account if necessary.

Related Account Notification

This section contains additional information about related account notifications shown in the Amazon SES reputation dashboard.

Why you received this notification

We have detected serious problems related to emails sent from another Amazon SES account. We believe that the problematic account is related to your AWS account, so we have taken action to avoid similar problems.

What you can do to resolve the issue

When we pause an account's ability to send email, we always send information about the reasons for the sending pause to the owner of that account. Refer to the email we sent to the owner of the related account for more information.
Spamtrap Notification

This section contains additional information about spamtrap notifications shown in the Amazon SES reputation dashboard.

Why you received this notification

A third-party anti-spam organization has reported to us that their spamtrap addresses recently received email from a verified address or domains associated with your Amazon SES account.

A spamtrap is a dormant email address that is used exclusively to lure unsolicited email (spam). A large number of spamtrap reports can have a negative impact on all Amazon SES users. To protect your reputation and that of other Amazon SES customers, we take immediate action when an account sends a particular volume of email to spamtrap addresses.

What you can do to resolve the issue

We can't reveal the email addresses associated with the spamtrap you encountered. These addresses are closely guarded by the organizations that own them, and once the addresses are known, they become worthless.

Sending email to spamtrap addresses typically indicates that there is an issue with how you acquire your customers' email addresses. For example, purchased lists of email addresses can contain spamtrap addresses, which is why sending to purchased or rented lists is prohibited by the Amazon SES terms of service. To acquire new addresses, we recommend that you implement a double opt-in strategy, as described in Building and maintaining your lists (p. 468). We recommend that you only send email to addresses that have completed the double opt-in process.

Additionally, you should purge your lists of addresses that haven't interacted with your emails recently. You can use open and click tracking, as described in Monitoring your Amazon SES sending activity (p. 259), to determine which users are viewing and interacting with the content you send.

If your account is under review

At the end of the review period, if messages are still being sent to spamtrap addresses from your account, we might pause your account's ability to send email until you resolve the issue.

If you have implemented changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. Provide detailed information about the steps you've taken to resolve the issue, and describe how these steps prevent the issue from happening again in the future. If we agree that the changes you've made appropriately address the issue, we cancel the review period or remove the sending pause from your account.

If your account's ability to send email is paused

You can request that we reconsider this decision. For more information, see Amazon SES Sending review process FAQs (p. 489).

When you have implemented changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. Include details of the actions you
have taken to resolve this issue, as well as details of your plans to ensure that this issue doesn't occur again. After we receive your request, we review the information that you provided and change the status of your account if necessary.

**Vulnerable Site Notification**

This section contains additional information about vulnerable site notifications shown in the Amazon SES reputation dashboard.

**Why you received this notification**

A comprehensive review has found that messages are being sent from your account that we don't believe you intended to send. These messages are highly likely to be flagged as spam by mailbox providers and recipients.

Most often in these situations, a third party is abusing a feature of your website to send unwanted email. For example, if your website contains an "email to a friend," "contact us," "invite a friend," or similar feature, a third party can use that feature to send unsolicited email.

**What you can do to resolve the issue**

First, identify features of your website or applications that might allow third parties to send emails using Amazon SES without your knowledge. In your Support Center case, you can request a sample of the messages we believe were sent in this manner.

Next, modify your application or website to prevent unsolicited sending. For example, add a CAPTCHA, limit the rate at which emails can be sent, remove the ability of users to submit custom content, require users to log in to send email, and remove the ability for the application to generate multiple simultaneous notifications.

**If your account is under review, or if your account's ability to send email is paused**

When you have implemented changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. Include details of the actions you have taken to resolve this issue, as well as details of your plans to ensure that this issue doesn't occur again. After we receive your request, we review the information that you provided and change the status of your account if necessary.

If we remove a review period or sending pause from your account, and we observe the same issue later, we might place your account under review or pause your ability to send email again. If we observe extreme issues or repeated instances of the same issue, we might permanently suspend your account's ability to send email.

See Amazon SES Sending review process FAQs (p. 489) for more information about what to do if your account is under review, or your account's ability to send email is paused.

**Other Notification**

This section contains additional information about other notifications shown in the Amazon SES reputation dashboard.

**Why you received this notification**

An automatic or human review has identified issues that aren't listed in the previous sections of this document.
What you can do to resolve the issue

Refer to the Support Center case that we opened on your behalf for details on the specific issue. To access Support Center, sign into the AWS Console and then choose Support Center. In your response to the case, describe the changes you implemented. Depending on your specific situation and the nature of the issues we discovered, we might end the review period or restore your account's ability to send email.

Creating reputation monitoring alarms using CloudWatch

Amazon SES automatically publishes a series of reputation-related metrics to Amazon CloudWatch. You can use these metrics to create alarms that notify you when your bounce or complaint rates reach levels that could impact your account's ability to send email.

Note
The procedures in this section omit some information about optional settings for CloudWatch alarms. For detailed instructions, see Creating Amazon CloudWatch Alarms in the Amazon CloudWatch User Guide.

To create a CloudWatch alarm

1. Create a new Amazon SNS topic, and then subscribe to it using your preferred endpoint (such as email or SMS). For more information, see Creating a Topic and Subscribing an Endpoint to a Topic in the Amazon Simple Notification Service Developer Guide.
2. Open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
3. In the navigation pane, choose Alarms.
4. On the Specify metric and conditions page, do the following:
   a. Under Metric, choose Select metric.
   b. In the list of metrics, choose SES.
      Note
      If you've never sent an email in the current AWS Region, SES might not appear in the list of available metrics. You can make the SES metrics appear by sending a test email to the Amazon SES mailbox simulator (p. 181). The metrics appear in CloudWatch within a few minutes.
   c. Choose Account Metrics.
   d. Choose the metric that you want to create the alarm for.
      For example, if you want to create an alarm when your bounce rate reaches a certain level, choose Reputation.BounceRate. If you want to create an alarm when your complaint rate reaches a certain level, choose Reputation.ComplaintRate.
      Note
      The Reputation.BounceRate and Reputation.ComplaintRate metrics won't appear on this page if your account has never had a bounce or a complaint, respectively.
   e. In the Conditions section, under Threshold type, choose Static.
   f. Under Whenever Reputation.MetricName is, choose Greater/Equal.
   g. Under than, specify the value that should cause CloudWatch to raise an alarm.
      If you're creating an alarm to monitor your bounce rate, note that Amazon SES recommends that you maintain a bounce rate under 5%. If the bounce rate for your account is greater than
10%, we might pause your account's ability to send email. For this reason, you should configure CloudWatch to send you a notification when the bounce rate for your account is greater than or equal to 0.05 (5%), as shown in the following image.

If you're creating an alarm to monitor your complaint rate, note that Amazon SES recommends that you maintain a complaint rate under 0.1%. If the complaint rate for your account is greater than 0.5%, we might pause your account's ability to send email. For this reason, you should configure CloudWatch to send you a notification when the complaint rate for your account is greater than or equal to 0.001 (0.1%), as shown in the following image.
Creating alarms using CloudWatch

5. On the **Configure actions** page, do the following:
   a. Under **Whenever this alarm state is**, choose **in Alarm**.
   b. Under **Select an SNS topic**, choose **Select an existing SNS topic**. For **Send notification to**, choose the topic that you created and subscribed to in step 1.
   c. Choose **Next**.

6. On the **Add a description** page, do the following:
   a. For **Alarm name**, enter a unique name for the alarm.
   b. (Optional) For **Alarm description**, enter some text that describes the alarm.
   c. Choose **Next**.

7. On the **Preview and create** page, confirm the settings that you specified on the preceding pages. When you're ready to create the alarm, choose **Create alarm**.
SNDS metrics for dedicated IPs

You can view Smart Network Data Services (SNDS) data for leased dedicated IP addresses in each AWS Region where you use Amazon SES. This SNDS data is available through the Amazon CloudWatch console.

SNDS is an Outlook program that allows IP owners to help prevent spam within their IP space. Amazon SES provides this important data for those who lease dedicated IPs. The SNDS data provides insight into the IP’s mail sending behavior and calls out areas of concern for your sender reputation.

Note
When referring to Outlook, this covers all the domains they track. For example, this can cover Hotmail.com, Outlook.com, and Live.com.

To view SNDS data for your dedicated IP addresses

2. In the navigation pane, choose Metrics.
3. On the All metrics tab in the AWS Region of your choice, choose SES under AWS Namespaces.
4. Choose IP Metrics, which will show you all of your dedicated IPs tracked by SNDS.
5. View all of your dedicated IPs tracked by SNDS in this list, or select an individual IP address to view only its metrics.

The following metrics are provided for each dedicated IP address and defined by Outlook. For more information, see Outlook’s SNDS FAQs.

Note
These metrics represent an activity period that provides updated data once a day. The metrics also have a corresponding timestamp, which reflects a 24-hour period.

- **SNDS.RCPTCommands** - This is the number of RCPT commands perceived by SNDS for the specific IP address during the activity period. RCPT commands are part of the SMTP protocol used to send mail, which specifies the recipient address to which you are trying to deliver email.
- **SNDS.DATACommands** - The number of DATA commands perceived by SNDS for the specific IP address during the activity period. DATA commands are part of the SMTP protocol used to send mail, specifically that part which actually transmits the message to the previously established intended recipient(s).
- **SNDS.MessageRecipients** - The number of recipients on messages perceived by SNDS for the specific IP address during the activity period.
- **SNDS.SpamRate** - Displays the aggregate results of the spam filtering applied to all messages sent by the IP address during the given activity period.
  - A SpamRate of 0 means the IP address has less than 10% spam.
  - A SpamRate of 0.5 means that between 10% and 90% spam is generated from the IP address.
  - A SpamRate of 1 means 90% or more spam is generated from the IP address.
- **SNDS.ComplaintRate** - This is the fraction of the time that a message received from the IP is complained about by an Outlook user during the activity period.
  - A ComplaintRate of 1 means a 100% complaint rate.
  - A ComplaintRate of 0.05 would mean a 5% complaint rate, for example.
  - A ComplaintRate of 0 means the rate is less than 0.1%.
- **SNDS.TrapHits** - Displays the number of messages sent to “trap accounts.” Trap accounts are accounts maintained by Outlook that don’t solicit any mail. Thus, any messages sent to trap accounts are very likely to be spam.
Troubleshooting questions

Q1. Why does data not populate every day? Either of the following scenarios could apply:

- SNDS data is dependent on Outlook’s SNDS program.
- There is a minimum threshold of emails SNDS needs to receive to calculate a value. Data may not be available at times where email volume on an IP was low.

Q2. Why are the SNDS.SpamRate and SNDS.ComplaintRate metrics changing, and what do I do if the rate changes to a value of 1?

This is an indicator that something in your sending behavior has triggered a negative response from the Outlook SNDS program. In this case, you want to check other Internet Service Providers (ISPs) as well as your engagement numbers to make sure it isn't a global problem. If it is a global problem, you may see issues with multiple ISPs, which would suggest a list, content, distribution, or permissions problem. If it is specific to Outlook, review how to best deliver to Outlook. For more information about best practices, see Best practices for sending email using Amazon SES (p. 465).

Q3. What actions will AWS Support take if my SNDS.SpamRate changes from a value of 0 (or 0.5) to 1?

AWS does not have any control over SNDS and therefore has no influence over SNDS. All mitigation requests need to filed directly with Outlook via their New support request form.

Automatically pausing email sending

To protect your sender reputation, you can temporarily pause email sending for messages sent using specific configuration sets, or for all messages sent from your Amazon SES account in a specific AWS Region.

By using Amazon CloudWatch and Lambda, you can create a solution that automatically pauses your email sending when your reputation metrics (such as bounce rate or complaint rate) exceed certain thresholds. This topic contains procedures for setting up this solution.

Topics in this section:
- Automatically pausing email sending for your entire Amazon SES account (p. 386)
- Automatically pausing email sending for a configuration set (p. 391)

Automatically pausing email sending for your entire Amazon SES account

The procedures in this section explain the steps to set up Amazon SES, Amazon SNS, Amazon CloudWatch, and AWS Lambda to automatically pause email sending for your Amazon SES account in a single AWS Region. If you send email from multiple regions, repeat the procedures in this section for each region in which you want to implement this solution.

Topics in this section:
- Part 1: Create an IAM Role (p. 387)
- Part 2: Create the Lambda Function (p. 387)
- Part 3: Re-Enable Email Sending for Your Account (p. 388)
- Part 4: Create an Amazon SNS Topic (p. 389)
Part 1: Create an IAM Role

The first step in configuring automatic pausing of email sending is to create an IAM role that can execute the UpdateAccountSendingEnabled API operation.

To create the IAM role

1. Open the IAM console at https://console.aws.amazon.com/iam/
2. In the navigation pane, choose Roles.
3. Choose Create role.
4. Under Select type of trusted entity, choose AWS service.
5. Under Choose the service that will use this role, choose Lambda. Choose Next: Permissions.
6. On the Attach permissions policies page, choose the following policies:
   - AWSLambdaBasicExecutionRole
   - AmazonSESFullAccess

   **Tip**
   Use the search box at the top of the list of policies to quickly locate these policies.

   Choose Next: Review.
7. On the Review page, for Name, type a name for the role. Choose Create role.

Part 2: Create the Lambda Function

After you create an IAM role, you can create the Lambda function that pauses email sending for your account.

To create the Lambda function

1. Open the AWS Lambda console at https://console.aws.amazon.com/lambda/.
2. Use the region selector to choose the region in which you want to deploy this Lambda function.

   **Note**
   This function only pauses email sending in the AWS Region you select in this step. If you send email from more than one region, repeat the procedures in this section for each region in which you want to automatically pause email sending.

3. Choose Create function.
4. Under Create function, choose Author from scratch.
5. Under Author from scratch, complete the following steps:
   - For Name, type a name for the Lambda function.
   - For Runtime, choose Node.js 6.10.
   - For Role, choose Choose an existing role.
   - For Existing role, choose the IAM role you created in the section called “Part 1: Create an IAM Role” (p. 387).

   Choose Create function.
6. Under **Function code**, in the code editor, paste the following code:

```javascript
'use strict';

var aws = require('aws-sdk');

// Create a new SES object.
var ses = new aws.SES();

// Specify the parameters for this operation. In this case, there is only one
// parameter to pass: the Enabled parameter, with a value of false
// (Enabled = false disables email sending, Enabled = true enables it).
var params = {
    Enabled: false
};

exports.handler = (event, context, callback) => {
    // Pause sending for your entire SES account
    ses.updateAccountSendingEnabled(params, function(err, data) {
        if(err) {
            console.log(err.message);
        } else {
            console.log(data);
        }
    });
};
```

Choose **Save**.

7. Choose **Test**. If the **Configure test event** window appears, type a name in the **Event name** field, and then choose **Create**.

8. Ensure that the notification bar at the top of the page says **Execution result: succeeded**. If the function failed to execute, do the following:

   - Verify that the IAM role you created in the section called **Part 1: Create an IAM Role** (p. 387) contains the correct policies.
   - Verify that the code in the Lambda function does not contain any errors. The Lambda code editor automatically highlights syntax errors and other potential issues.

---

**Part 3: Re-Enable Email Sending for Your Account**

A side effect of testing the Lambda function in the section called "Part 2: Create the Lambda Function" (p. 387) is that email sending for your Amazon SES account is paused. In most cases, you do not want to pause sending for your account until the CloudWatch alarm is triggered.

The procedures in this section re-enable email sending for your Amazon SES account. To complete these procedures, you must install and configure the AWS Command Line Interface. For more information, see the **AWS Command Line Interface User Guide**.

**To re-enable email sending**

1. At the command line, type the following command to re-enable email sending for your account: `aws ses update-account-sending-enabled --enabled --region us-west-2`

   **Note**
   
   Replace `us-west-2` in the preceding command with the name of the region in which you want to re-enable email sending.

2. At the command line, type the following command to check the email sending status for your account: `aws ses get-account-sending-enabled --region us-west-2`
If you see the following output, then you have successfully re-enabled email sending for your account:

```
{
   "Enabled": true
}
```

### Part 4: Create an Amazon SNS Topic

For CloudWatch to execute your Lambda function when an alarm is triggered, you must first create an Amazon SNS topic and subscribe the Lambda function to it.

**To create the Amazon SNS topic**

2. Use the region selector to choose the region in which you want to automatically pause email sending.
3. In the navigation pane, choose Topics.
4. Choose Create new topic.
5. On the Create new topic window, for Topic name, type a name for the topic. Optionally, you can type a more descriptive name in the Display name field.
6. Choose Create topic.
7. In the list of topics, check the box next to the topic you created in the previous step. On the Actions menu, choose Subscribe to topic.
8. On the Create subscription window, make the following selections:
   - For Protocol, choose AWS Lambda.
   - For Endpoint, choose the Lambda function you created in the section called “Part 2: Create the Lambda Function” (p. 387).
   - For Version or alias, choose default.
9. Choose Create subscription.

### Part 5: Create a CloudWatch Alarm

This section contains procedures for creating an alarm in CloudWatch that is triggered when a metric reaches a certain threshold. When the alarm is triggered, it delivers a notification to the Amazon SNS topic you created in the section called “Part 4: Create an Amazon SNS Topic” (p. 389), which then executes the Lambda function you created in the section called “Part 2: Create the Lambda Function” (p. 387).

**To create a CloudWatch alarm**

2. Use the region selector to choose the region in which you want to automatically pause email sending.
3. In the navigation pane, choose Alarms.
4. Choose Create Alarm.
5. On the Create Alarm window, under SES Metrics, choose Account Metrics.
6. Under Metric Name, choose one of the following options:
Choose Next.

7. Complete the following steps:

- Under **Alarm Threshold**, for **Name**, type a name for the alarm.

- Under **Whenever**: **Reputation.BounceRate** or **Whenever**: **Reputation.ComplaintRate**, specify the threshold that causes the alarm to trigger.

  **Note**
  
  Your account is automatically placed under review if your bounce rate exceeds 10%, or if your complaint rate exceeds 0.5%. When you specify the bounce or complaint rate that causes the CloudWatch alarm to trigger, we recommend that you use values that are below these rates to prevent your account from being placed under review.

- Under **Actions**, for **Whenever this alarm**, choose **State is ALARM**. For **Send notification to**, choose the Amazon SNS topic you created in the section called “Part 4: Create an Amazon SNS Topic” (p. 389).

Choose **Create Alarm**.

### Part 6: Test the solution

You can now test the alarm to ensure that it executes the Lambda function when it enters the ALARM state. You can use the `SetAlarmState` API operation to temporarily change the state of the alarm.

The procedures in this section are optional, but we recommend that you complete them to ensure that the entire solution is configured correctly.

1. At the command line, type the following command to check the email sending status for your account: `aws ses get-account-sending-enabled --region us-west-2`

   **Note**
   
   Replace `us-west-2` in the preceding command with the name of the region you specified in the previous step.

   If sending is enabled for your account, you see the following output:

   ```json
   {
     "Enabled": true
   }
   ```

2. At the command line, type the following command to temporarily change the alarm state to ALARM: `aws cloudwatch set-alarm-state --alarm-name MyAlarm --state-value ALARM --state-reason "Testing execution of Lambda function" --region us-west-2`

   Replace `MyAlarm` in the preceding command with the name of the alarm you created in the section called “Part 5: Create a CloudWatch Alarm” (p. 389), and replace `us-west-2` with the region in which you want to automatically pause email sending.
Note
When you execute this command, the status of the alarm switches from OK to ALARM and back to OK within a few seconds. You can view these status changes on the alarm's History tab in the CloudWatch console, or by using the DescribeAlarmHistory operation.

3. At the command line, type the following command to check the email sending status for your account:

```
aws ses get-account-sending-enabled --region us-west-2
```

If the Lambda function executed successfully, you see the following output:

```
{
   "Enabled": false
}
```

4. Complete the steps in the section called “Part 3: Re-Enable Email Sending for Your Account” (p. 388) to re-enable email sending for your account.

Automatically pausing email sending for a configuration set

You can configure Amazon SES to export reputation metrics that are specific to emails that are sent using a specific configuration set to Amazon CloudWatch. You can then use these metrics to create CloudWatch alarms that are specific to these configuration sets. When these alarms exceed certain thresholds, you can automatically pause the sending of emails that use the specified configuration sets, without impacting the overall email sending capabilities of your Amazon SES account.

Note
The solution described in this section pauses email sending for a specific configuration set in a single AWS Region. If you send email from multiple regions, repeat the procedures in this section for each region in which you want to implement this solution.

Topics in this section:
- Part 1: Enable Reputation Metric Reporting for the Configuration Set (p. 391)
- Part 2: Create an IAM Role (p. 391)
- Part 3: Create the Lambda Function (p. 392)
- Part 4: Re-Enable Email Sending for the Configuration Set (p. 393)
- Part 5: Create an Amazon SNS Topic (p. 394)
- Part 6: Create a CloudWatch Alarm (p. 394)
- Part 7: Test the solution (p. 395)
To create the IAM role

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose Roles.
3. Choose Create role.
4. Under Select type of trusted entity, choose AWS service.
5. Under Choose the service that will use this role, choose Lambda. Choose Next: Permissions.
6. On the Attach permissions policies page, choose the following policies:
   - AWS LambdaBasicExecutionRole
   - AmazonSESFullAccess

   **Tip**
   Use the search box at the top of the list of policies to quickly locate these policies.

   Choose Next: Review.
7. On the Review page, for Name, type a name for the role. Choose Create role.

Part 3: Create the Lambda Function

After you create an IAM role, you can create the Lambda function that pauses email sending for the configuration set.

To create the Lambda function

1. Open the AWS Lambda console at https://console.aws.amazon.com/lambda/.
2. Use the region selector to choose the region in which you want to deploy this Lambda function.
   **Note**
   This function only pauses email sending for configuration sets in the AWS Region you select in this step. If you send email from more than one region, repeat the procedures in this section for each region in which you want to automatically pause email sending.
3. Choose Create function.
4. Under Create function, choose Author from scratch.
5. Under Author from scratch, complete the following steps:
   - For Name, type a name for the Lambda function.
   - For Runtime, choose Node.js 6.10.
   - For Role, choose Choose an existing role.
   - For Existing role, choose the IAM role you created in the section called “Part 2: Create an IAM Role” (p. 391).

   Choose Create function.
6. Under Function code, in the code editor, paste the following code:

```javascript
'use strict';

var aws = require('aws-sdk');

// Create a new SES object.
var ses = new aws.SES();
```
// Specify the parameters for this operation. In this example, you pass the
// Enabled parameter, with a value of false (Enabled = false disables email
// sending, Enabled = true enables it). You also pass the ConfigurationSetName
// parameter, with a value equal to the name of the configuration set for
// which you want to pause email sending.
var params = {
  ConfigurationSetName: ConfigSet,
  Enabled: false
};

exports.handler = (event, context, callback) => {
  // Pause sending for a configuration set
  ses.updateConfigurationSetSendingEnabled(params, function(err, data) {
    if(err) {
      console.log(err.message);
    } else {
      console.log(data);
    }
  });
};

Replace ConfigSet in the preceding code with the name of the configuration set. Choose Save.

7. Choose Test. If the Configure test event window appears, type a name in the Event name field, and then choose Create.

8. Ensure that the notification bar at the top of the page says Execution result: succeeded. If the function failed to execute, do the following:
   • Verify that the IAM role you created in the section called “Part 2: Create an IAM Role” (p. 391) contains the correct policies.
   • Verify that the code in the Lambda function does not contain any errors. The Lambda code editor automatically highlights syntax errors and other potential issues.

Part 4: Re-Enable Email Sending for the Configuration Set

A side effect of testing the Lambda function in the section called "Part 3: Create the Lambda Function" (p. 392) is that email sending for the configuration set is paused. In most cases, you do not want to pause sending for the configuration set until the CloudWatch alarm is triggered.

The procedures in this section re-enable email sending for your configuration set. To complete these procedures, you must install and configure the AWS Command Line Interface. For more information, see the AWS Command Line Interface User Guide.

To re-enable email sending

1. At the command line, type the following command to re-enable email sending for the configuration set:

```bash
aws ses update-configuration-set-sending-enabled \
  --configuration-set-name ConfigSet \
  --enabled
```

In the preceding command, replace ConfigSet with the name of the configuration set for which you want to pause email sending, and replace us-west-2 with the region in which you want to automatically pause email sending.

2. At the command line, type the following command to ensure that email sending is enabled:

```bash
aws ses describe-configuration-set \
  --configuration-set-name ConfigSet \
```


For a configuration set

```
--configuration-set-attribute-names reputationOptions
```

The command produces output that resembles the following example:

```
{
   "ConfigurationSet": {
      "Name": "ConfigSet"
   },
   "ReputationOptions": {
      "ReputationMetricsEnabled": true,
      "SendingEnabled": true
   }
}
```

If the value of `SendingEnabled` is `true`, then email sending for the configuration set was successfully re-enabled.

Part 5: Create an Amazon SNS Topic

For CloudWatch to execute the Lambda function when an alarm is triggered, you must first create an Amazon SNS topic and subscribe the Lambda function to it.

To create the Amazon SNS topic

2. Use the region selector to choose the region in which you want to automatically pause email sending.
3. In the navigation pane, choose Topics.
4. Choose Create new topic.
5. On the Create new topic window, for Topic name, type a name for the topic. Optionally, you can type a more descriptive name in the Display name field.
   Choose Create topic.
6. In the list of topics, check the box next to the topic you created in the previous step. On the Actions menu, choose Subscribe to topic.
7. On the Create subscription window, make the following selections:

   - For Protocol, choose AWS Lambda.
   - For Endpoint, choose the Lambda function you created in the section called “Part 3: Create the Lambda Function” (p. 392).
   - For Version or alias, choose default.
8. Choose Create subscription.

Part 6: Create a CloudWatch Alarm

This section contains procedures for creating an alarm in CloudWatch that is triggered when a metric reaches a certain threshold. When the alarm is triggered, it delivers a notification to the Amazon SNS topic you created in the section called “Part 5: Create an Amazon SNS Topic” (p. 394), which then executes the Lambda function you created in the section called “Part 3: Create the Lambda Function” (p. 392).

To create a CloudWatch alarm

2. Use the region selector to choose the region in which you want to automatically pause email sending.
3. In the navigation pane on the left, choose Alarms.
4. Choose Create Alarm.
5. On the Create Alarm window, under SES Metrics, choose Configuration Set Metrics.
6. In the ses:configuration-set column, locate the configuration set for which you want to create an alarm. Under Metric Name, choose one of the following options:
   - Reputation.BounceRate – Choose this metric if you want to pause email sending for the configuration set when the overall hard bounce rate for the configuration set crosses a threshold that you define.
   - Reputation.ComplaintRate – Choose this metric if you want to pause email sending for the configuration set when the overall complaint rate for the configuration set crosses a threshold that you define.
Choose Next.
7. Complete the following steps:
   - Under Alarm Threshold, for Name, type a name for the alarm.
   - Under Whenever: Reputation.BounceRate or Whenever: Reputation.ComplaintRate, specify the threshold that causes the alarm to trigger.
     **Note**
     If the overall bounce rate for your Amazon SES account exceeds 10%, or if the overall complaint rate for your Amazon SES account exceeds .5%, your Amazon SES account is automatically placed under review. When you specify the bounce or complaint rate that causes the CloudWatch alarm to trigger, we recommend that you use values that are far below these rates to prevent your account from being placed under review.
   - Under Actions, for Whenever this alarm, choose State is ALARM. For Send notification to, choose the Amazon SNS topic you created in the section called “Part 5: Create an Amazon SNS Topic“ (p. 394).
Choose Create Alarm.

**Part 7: Test the solution**

You can now test the alarm to ensure that it executes the Lambda function when it enters the ALARM state. You can use the SetAlarmState operation in the CloudWatch API to temporarily change the state of the alarm.

The procedures in this section are optional, but we recommend that you complete them to verify that the entire solution is configured correctly.

**To test the solution**

1. At the command line, type the following command to check the email sending status for the configuration set: `aws ses describe-configuration-set --configuration-set-name ConfigSet --region us-west-2`

   If sending is enabled for the configuration set, you see the following output:

   ```json
   {   "ConfigurationSet": {   "Name": "ConfigSet"   }
   ```
For a configuration set

```
},
  "ReputationOptions": {
    "ReputationMetricsEnabled": true,
    "SendingEnabled": true
  }
}
```

If the value of `SendingEnabled` is true, then email sending is currently enabled for the configuration set.

2. At the command line, type the following command to temporarily change the alarm state to ALARM:

```
aws cloudwatch set-alarm-state \
  --alarm-name MyAlarm \ 
  --state-value ALARM \ 
  --state-reason "Testing execution of Lambda function"
```

Replace `MyAlarm` in the preceding command with the name of the alarm you created in the section called "Part 6: Create a CloudWatch Alarm" (p. 394).

**Note**

When you execute this command, the status of the alarm switches from OK to ALARM and back to OK within a few seconds. You can view these status changes on the alarm’s History tab in the CloudWatch console, or by using the DescribeAlarmHistory operation.

3. At the command line, type the following command to check the email sending status for the configuration set:

```
aws ses describe-configuration-set \
  --configuration-set-name ConfigSet
```

If the Lambda function executed successfully, you see output that resembles the following example:

```
{
  "ConfigurationSet": {
    "Name": "ConfigSet"
  },
  "ReputationOptions": {
    "ReputationMetricsEnabled": true,
    "SendingEnabled": false
  }
}
```

If the value of `SendingEnabled` is false, then email sending for the configuration set is disabled, indicating that the Lambda function executed successfully.

4. Complete the steps in the section called "Part 4: Re-Enable Email Sending for the Configuration Set" (p. 393) to re-enable email sending for the configuration set.
Controlling access to Amazon SES

You can use AWS Identity and Access Management (IAM) with Amazon Simple Email Service (Amazon SES) to specify which Amazon SES API actions an IAM user, group, or role can perform. (In this topic we refer to these entities collectively as user.) You can also control which email addresses the user can use for the "From", recipient, and "Return-Path" addresses of emails.

For example, you can create an IAM policy that allows users in your organization to send email, but not perform administrative actions such as checking sending statistics. As another example, you can write a policy that allows a user to send emails through Amazon SES from your account, but only if they use a specific "From" address.

To use IAM, you define an IAM policy, which is a document that explicitly defines permissions, and attach the policy to a user. To learn how to create IAM policies, see the IAM User Guide. Other than applying the restrictions you set in your policy, there are no changes to how users interact with Amazon SES or in how Amazon SES carries out requests.

**Note**
You can also control access to Amazon SES by using sending authorization policies. Whereas IAM policies constrain what individual IAM users can do, sending authorization policies constrain how individual verified identities can be used. Further, only sending authorization policies can grant cross-account access. For more information about sending authorization, see Using sending authorization with Amazon SES (p. 149).

If you are looking for information about how to generate Amazon SES SMTP credentials for an existing IAM user, see Obtaining your Amazon SES SMTP credentials (p. 86).

Creating IAM Policies for Access to Amazon SES

This section explains how you can use IAM policies specifically with Amazon SES. To learn how to create IAM policies in general, see the IAM User Guide.

There are three reasons you might use IAM with Amazon SES:

- To restrict the email-sending action.
- To restrict the "From", recipient, and "Return-Path" addresses of the emails that the user sends.
- To control general aspects of API usage such as the time period during which a user is permitted to call the APIs that they are authorized to use.

Restricting the Action

To control which Amazon SES actions a user can perform, you use the Action element of an IAM policy. You can set the Action element to any Amazon SES API action by prefixing the API name with the lowercase string ses:. For example, you can set the Action to ses:SendEmail, ses:GetSendStatistics, or ses:* (for all actions).

Then, depending on the Action, specify the Resource element as follows:

**If the Action element only permits access to email-sending APIs (that is, ses:SendEmail and/or ses:SendRawEmail):**

- To allow the user to send from any identity in your AWS account, set Resource to *
• To restrict the identities that a user is allowed to send from, set Resource to the ARNs of the identities that you are permitting the user to use.

**If the Action element permits access to all APIs:**

• If you don’t want to restrict the identities that the user can send from, set Resource to *
• If you want to restrict the identities that a user is allowed to send from, you need to create two policies (or two statements within one policy):
  • One with Action set to an explicit list of the permitted non-email-sending APIs and Resource set to *
  • One with Action set to one of the email-sending APIs (ses:SendEmail and/or ses:SendRawEmail), and Resource set to the ARN(s) of the identities you are permitting the user to use.

For a list of available Amazon SES actions, see the Amazon Simple Email Service API Reference and Amazon Simple Email Service API v2 Reference. If the IAM user will be using the SMTP interface, you must allow access to ses:SendRawEmail at a minimum.

## Restricting Email Addresses

If you want to restrict the user to specific email addresses, you can use a Condition block. In the Condition block, you specify conditions by using condition keys as described in the IAM User Guide. By using condition keys, you can control the following email addresses:

**Note**
These email address condition keys apply only to the APIs noted in the following table.

<table>
<thead>
<tr>
<th>Condition Key</th>
<th>Description</th>
<th>API</th>
</tr>
</thead>
<tbody>
<tr>
<td>ses:Recipients</td>
<td>Restricts the recipient addresses, which include the To:, “CC”, and “BCC” addresses.</td>
<td>SendEmail, SendRawEmail</td>
</tr>
<tr>
<td>ses:FromAddress</td>
<td>Restricts the “From” address.</td>
<td>SendEmail, SendRawEmail, SendBounce</td>
</tr>
<tr>
<td>ses:FromDisplayName</td>
<td>Restricts the “From” address that is used as the display name.</td>
<td>SendEmail, SendRawEmail</td>
</tr>
<tr>
<td>ses:FeedbackAddress</td>
<td>Restricts the “Return-Path” address, which is the address where bounces and complaints can be sent to you by email feedback forwarding. For information about email feedback forwarding, see Amazon SES notifications sent by email (p. 265).</td>
<td>SendEmail, SendRawEmail</td>
</tr>
</tbody>
</table>

## Restricting General API Usage

By using AWS-wide keys in conditions, you can restrict access to Amazon SES based on aspects such as the date and time that a user is permitted access to APIs. Amazon SES implements only the following AWS-wide policy keys:
Example IAM Policies for Amazon SES

This topic provides examples of policies that permit a user access to Amazon SES, but only under certain conditions.

Policy examples in this section:
- Allowing Full Access to All Amazon SES Actions (p. 399)
- Allowing Access to Email-Sending Actions Only (p. 399)
- Restricting the Time Period of Sending (p. 400)
- Restricting the Recipient Addresses (p. 400)
- Restricting the "From" Address (p. 401)
- Restricting the Display Name of the Email Sender (p. 401)
- Restricting the Destination of Bounce and Complaint Feedback (p. 402)

Allowing Full Access to All Amazon SES Actions

The following policy allows a user to call any Amazon SES action.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ses:*"
      ],
      "Resource": "*"
    }
  ]
}
```

Allowing Access to Email-Sending Actions Only

The following policy permits a user to send email using Amazon SES, but does not permit the user to perform administrative actions such as accessing Amazon SES sending statistics.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
```
Restricting the Time Period of Sending

The following policy permits a user to call Amazon SES email-sending APIs only during the month of September 2018.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Resource": "*",
      "Condition": {
        "DateGreaterThan": {
          "aws:CurrentTime": "2018-08-31T12:00Z"
        },
        "DateLessThan": {
          "aws:CurrentTime": "2018-10-01T12:00Z"
        }
      }
    }
  ]
}
```

Restricting the Recipient Addresses

The following policy permits a user to call the Amazon SES email-sending APIs, but only to recipient addresses in domain `example.com`.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Resource": "*",
      "Condition": {
        "ForAllValues:StringLike": {
          "ses:Recipients": [
            "*@example.com"
          ]
        }
      }
    }
  ]
}
```
Restricting the "From" Address

The following policy permits a user to call the Amazon SES email-sending APIs, but only if the "From" address is marketing@example.com.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action": ["ses:SendEmail", "ses:SendRawEmail" ],
         "Resource":"*",
         "Condition":{
            "StringEquals":{
               "ses:FromAddress":"marketing@example.com"
            }
         }
      }
   ]
}
```

The following policy permits a user to call the SendBounce API, but only if the "From" address is bounce@example.com.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action": ["ses:SendBounce" ],
         "Resource":"*",
         "Condition":{
            "StringEquals":{
               "ses:FromAddress":"bounce@example.com"
            }
         }
      }
   ]
}
```

Restricting the Display Name of the Email Sender

The following policy permits a user to call the Amazon SES email-sending APIs, but only if the display name of the "From" address includes Marketing.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action": ["ses:SendEmail", "ses:SendRawEmail" ],
         "Resource":"*",
```
Restricting the Destination of Bounce and Complaint Feedback

The following policy permits a user to call the Amazon SES email-sending APIs, but only if the "Return-Path" of the email is set to feedback@example.com.

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "ses:SendEmail",
        "ses:SendRawEmail"
      ],
      "Resource":"*",
      "Condition":{
        "StringEquals":{
          "ses:FeedbackAddress":"feedback@example.com"
        }
      }
    }
  ]
}
```
Logging Amazon SES API calls with AWS CloudTrail

Amazon SES is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in Amazon SES. CloudTrail captures API calls for Amazon SES as events. The calls captured include calls from the Amazon SES console and code calls to the Amazon SES API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for Amazon SES. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in Event history. Using the information collected by CloudTrail, you can determine the request that was made to Amazon SES, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, including how to configure and enable it, see the AWS CloudTrail User Guide.

Amazon SES Information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When supported event activity occurs in Amazon SES, that activity is recorded in a CloudTrail event along with other AWS service events in Event history. You can view, search, and download recent events in your AWS account. For more information, see Viewing Events with CloudTrail Event History.

For an ongoing record of events in your AWS account, including events for Amazon SES, create a trail. A trail enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following:

- Overview for Creating a Trail
- CloudTrail Supported Services and Integrations
- Configuring Amazon SNS Notifications for CloudTrail
- Receiving CloudTrail Log Files from Multiple Regions and Receiving CloudTrail Log Files from Multiple Accounts

Amazon SES supports logging the following actions as events in CloudTrail log files:

- CloneReceiptRuleSet
- CreateReceiptFilter
- CreateReceiptRule
- CreateReceiptRuleSet
- DeleteIdentity
- DeleteIdentityPolicy
- DeleteReceiptFilter
- DeleteReceiptRule
- DeleteReceiptRuleSet
• DeleteVerifiedEmailAddress
• DescribeActiveReceiptRuleSet
• DescribeReceiptRule
• DescribeReceiptRuleSet
• GetIdentityDkimAttributes
• GetIdentityNotificationAttributes
• GetIdentityPolicies
• GetIdentityVerificationAttributes
• GetSendQuota
• GetSendStatistics
• ListIdentities
• ListIdentityPolicies
• ListReceiptFilters
• ListReceiptRuleSets
• ListVerifiedEmailAddresses
• PutIdentityPolicy
• ReorderReceiptRuleSet
• SetActiveReceiptRuleSet
• SetReceiptRulePosition
• SetIdentityDkimEnabled
• SetIdentityFeedbackForwardingEnabled
• SetIdentityHeadersInNotificationsEnabled
• SetIdentityNotificationTopic
• UpdateReceiptRule
• VerifyDomainDkim
• VerifyDomainIdentity
• VerifyEmailAddress
• VerifyEmailIdentity

Note
Amazon SES delivers management events to CloudTrail. Management events include actions that are related to creating and managing resources within your AWS account. In Amazon SES, management events include actions such as creating and deleting identities or receipt rules. Management events are different from data events. Data events are events that are related to accessing and interacting with data within your AWS account. In Amazon SES, data events include actions such as sending emails.

Because Amazon SES only delivers management events to CloudTrail, the following events aren't recorded in CloudTrail:

• SendEmail
• SendRawEmail
• SendTemplatedEmail
• SendBulkTemplatedEmail
• SendCustomVerificationEmail

You can use event publishing to record events related to email sending. For more information, see Monitor email sending using Amazon SES event publishing (p. 288).
Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or AWS Identity and Access Management (IAM) user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the CloudTrail userIdentity Element.

Example: Amazon SES Log File Entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The following example shows a CloudTrail log entry that demonstrates the DeleteIdentity and VerifyEmailIdentity actions.

```json
{
  "Records": [  
    {
      "awsRegion": "us-west-2",
      "eventID": "offa308d-1467-4259-8be3-c749753be325",
      "eventName": "DeleteIdentity",
      "eventSource": "ses.amazonaws.com",
      "eventTime": "2018-02-02T21:34:50Z",
      "eventType": "AwsApiCall",
      "eventVersion": "1.02",
      "recipientAccountId": "111122223333",
      "requestID": "50b87bfe-ab23-11e4-9106-5b36376f9d12",
      "requestParameters": {
        "identity": "amazon.com"
      },
      "responseElements": null,
      "sourceIPAddr": "192.0.2.0",
      "userIdentity": {
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "accountId": "111122223333",
        "arn": "arn:aws:iam::111122223333:root",
        "principalId": "111122223333",
        "type": "Root"
      }
    },
    {
      "awsRegion": "us-west-2",
      "eventID": "5653b0ff-8556-4526-9b53-a603a9231725",
      "eventName": "VerifyEmailIdentity",
      "eventSource": "ses.amazonaws.com",
      "eventTime": "2018-02-04T01:03:32Z",
      "eventType": "AwsApiCall",
      "eventVersion": "1.02",
      "recipientAccountId": "111122223333",
      "requestID": "eb2ff003-ac09-11e4-8ff5-a56a3119e253",
      "requestParameters": {
        "emailAddress": "sender@example.com"
      }
    }
  ]
}
```
Example: Amazon SES Log File Entries

```
},
"responseElements":null,
"sourceIPAddress":"192.0.2.0",
"userAgent":"aws-sdk-java/unknown-version",
"userIdentity":{
  "accessKeyId":"AKIAIOSFODNN7EXAMPLE",
  "accountId":"111122223333",
  "arn":"arn:aws:iam::111122223333:root",
  "principalId":"111122223333",
  "type":"Root"
}
]
}
```
Using the Amazon SES API

You can access the Amazon SES API using an AWS SDK, which wraps the low-level functionality of the Amazon SES API with higher-level data types and function calls that take care of the details for you. You can also make HTTPS requests to the Amazon SES API. You can find additional information about individual API operations in the Amazon Simple Email Service API Reference.

This section contains the following topics:
- Amazon SES API requests (p. 407)
- Authenticating requests to the Amazon SES API (p. 409)
- GET and POST examples for the Amazon SES API (p. 410)
- Amazon SES API responses (p. 411)
- Error codes returned by the Amazon SES API (p. 413)

Amazon SES API requests

You can call the Amazon SES API directly by making requests to an Amazon SES API endpoint. Requests to the Amazon SES API are simple HTTPS requests that use the GET or POST method. API requests have to contain an Action parameter to indicate the action to be performed.

Important
Amazon SES doesn't support HTTP requests. You have to use HTTPS instead.

Structure of a GET request

This guide presents the Amazon SES GET requests as URLs. Each URL consists of the following:
- **Endpoint**—The resource the request is acting on. For a list of endpoint URLs for the AWS Regions where Amazon SES is available, see Amazon Simple Email Service (Amazon SES) in the AWS General Reference.
- **Action**—The action you want to perform on the endpoint, such as sending a message.
- **Parameters**—Any request parameters.

The following is an example GET request to send a message using the Amazon SES endpoint in the US West (Oregon) region.

```
https://email.us-west-2.amazonaws.com?Action=SendEmail&Source=user%40example.com&Destination.ToAddresses.member.1=allan%40example.com&Message.Subject.Data=This%20is%20the%20subject&Message.Body.Text.Data=Hello.%20I%20hope%20you%20are%20having%20a%20good%20day.
```

Important
Because the GET requests are URLs, you must URL-encode the parameter values. For example, in the preceding example request, the value for the Source parameter is actually user@example.com. However, the "@" character is not allowed in URLs, so each "@" is URL-encoded as "%40".

To make the GET examples easier to read, this guide presents them in the following parsed format.
The first line represents the **endpoint** of the request. After the endpoint is a question mark (?), which separates the endpoint from the parameters. Each parameter is separated by an ampersand (&).

The **Action** parameter indicates the action to perform. For a complete list of actions, and the parameters used with each action, see the [Amazon Simple Email Service API Reference](https://docs.aws.amazon.com/ses/latest/api/index.html).

Some operations take lists of parameters. For example, when you send an email to multiple recipients, you can provide a list of email addresses. You specify this type of list with `param.n` notation, where values of `n` are integers starting from 1. For example, you would specify the first "To:" address using `Destination.ToAddresses.1`, the second with `Destination.ToAddresses.2`, etc.

In Amazon SES, spaces are not allowed in any of the parameter values. In this guide, any example request parameter value that includes spaces is displayed in one of two different ways:

- URL-encoded (as `%20`).
- Represented by a plus sign (“+”). Within a request, a plus sign is reserved as a shorthand notation for a space. (If you want to include a literal, uninterpreted plus sign in any parameter, you must URL-encode it as `%2B`.)

**Note**

Every request must be accompanied by an `X-Amzn-Authorization` HTTP header. For more information, see Authenticating requests to the Amazon SES API Reference (p. 409).

### Structure of a POST request

Amazon SES also accepts POST requests. With a POST request, you send the parameters as a form in the HTTP request body as described in the following procedure.

#### To create a POST request

1. Assemble the parameter names and values into a form.

   Put the parameters and values together as you would for a GET request (with an ampersand separating each name-value pair). The following example shows a `SendEmail` request with the line breaks we use in this guide to make the information easier to read.

   ```
   Action=SendEmail
   &Source=user@example.com
   &Destination.ToAddresses.member.1=allan@example.com
   &Message.Subject.Data=This is the subject line.
   &Message.Body.Text.Data=Hello. I hope you are having a good day.
   ```

2. Form-URL-encode the form according to the Form Submission section of the HTML specification.

   For more information, see [http://www.w3.org/MarkUp/html-spec/html-spec_toc.html#SEC8.2.1](http://www.w3.org/MarkUp/html-spec/html-spec_toc.html#SEC8.2.1).

   ```
   Action=SendEmail
   &Source=user@example.com
   &Destination.ToAddresses.member.1=allan@example.com
   &Message.Subject.Data=This%20is%20the%20subject%20line.
   ```
3. Provide the resulting form as the body of the POST request.
4. Include the following HTTP headers in the request:
   - `Content-Type`, with the value set to `application/x-www-form-urlencoded`
   - `Content-Length`
   - `Date`
   - `Authorization` (For more information, see Authenticating requests to the Amazon SES API (p. 409).)
5. Send the completed request.

```
POST / HTTP/1.1
Date: Thu, 26 May 2011 06:49:50 GMT
Host: email.us-west-2.amazonaws.com
Content-Type: application/x-www-form-urlencoded
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE,SignedHeaders=Date;x-amz-date,Signature=9d63c3b5b7623d1fa3dc7fd1547313b9546c6d0fbb6773a420613b7EXAMPLE
Content-Length: 230

Action=SendEmail
&Source=user%40example.com
&Destination.ToAddresses.member.1=allan%40example.com
&Message.Subject.Data=This%20is%20the%20subject%20line.
```

**Note**
Your HTTP client typically adds other items to the HTTP request as required by the version of HTTP that the client uses. We don't include those additional items in the examples in this guide.

### Authenticating requests to the Amazon SES API

When you access the Amazon SES API, you authenticate your request using the AWS Signature. If your request doesn't include a valid signature, Amazon SES returns an error and doesn't process the request.

For more information about signing AWS requests, see Signing AWS requests in the AWS General Reference.

**Important**
Beginning October 1st, 2020, Amazon SES will only support requests signed using Signature Version 4. If you use an older version of the AWS Signature, you must adopt AWS Signature Version 4 prior to that date.

### Authentication errors

If you attempt to send email, and your request uses an earlier version of the AWS Signature, you see an error message that resembles the following example:

```
<ErrorResponse xmlns="http://ses.amazonaws.com/doc/2010-12-01/"><Error><Type>Sender</Type><Code>InvalidClientTokenId</Code><Message>The security token included in the request is invalid</Message></Error></ErrorResponse>
```
You can receive this error if you use an older version of an AWS SDK or the AWS CLI. You can also see this error if you make a direct HTTPS request to the Amazon SES API that uses an older version of the AWS Signature.

If you receive this error message, you have to update your requests to use the AWS Signature Version 4 by replacing your existing SMTP credentials using the appropriate procedure relative to your setup:

- If you generated your SMTP credentials using the SES Console, simply create new credentials and replace your existing credentials with the new ones.
- If you derived your SMTP credentials from your AWS credentials, make sure you are using the AWS Signature Version 4 algorithm. If you rely on a library to do this conversion, check if the library has a newer release that uses the AWS Signature Version 4 algorithm and migrate to it. Otherwise, you'll need to either derive the credentials from another library that uses AWS Signature Version 4 algorithm or generate credentials using the SES console.

Migrating to the AWS Signature Version 4

If you use an AWS SDK or the AWS CLI, you should update to the most recent version of the SDK or the AWS CLI. If you make direct HTTPS requests to the Amazon SES API, update the headers in your requests to use AWS Signature Version 4. You can easily identify API requests that use earlier versions of the AWS Signature by looking the request headers. For example, requests that use the AWS Signature Version 3 resemble the following example.

```
X-Amzn-Authorization: AWS3-HTTPS
AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE,Algorithm=HMACSHA256,Signature=lBP67vCvGl ...
```

Requests that use the AWS Signature Version 4 include an Authorization header that contains the following information:

- The algorithm you used for signing (AWS4-HMAC-SHA256)
- The credential scope (with your access key ID)
- A list of signed headers
- The calculated signature. The signature is based on your request information, and you use your AWS secret access key to produce the signature. The signature confirms your identity to AWS.

You can find an example of a call to the Amazon SES API that uses the AWS Signature Version 4 in Amazon SES API requests (p. 407). For more information about using the AWS Signature Version 4, see Signature Version 4 signing process in the AWS General Reference.

GET and POST examples for the Amazon SES API

This section contains examples of requests that you can issue against the Amazon SES API.

Example GET request

The following code is an example of a GET request. It includes a calculated signature. Notice that all of the parameters in the request are URL-encoded.

```
https://email.us-west-2.amazonaws.com/?Action=SendEmail
```
Example POST request

The following code is an example of a POST request. It includes a calculated signature. Notice that all of the parameters in the request are URL-encoded.

```
POST / HTTP/1.1
Host: email.us-west-2.amazonaws.com
Content-Type: application/x-www-form-urlencoded
Date: Tue, 25 May 2010 21:20:27 +0000
Content-Length: 174

Action=SendRawEmail
&Destinations.member.1=allan%40example.com
&RawMessage.Data=RnJvbTp1c2VyQGV4YW1wbGUuY29tDQpTdWJqZWN0OiBUZXN0DQoNCk1lc3 ...

The value for RawMessage.Data is a base64-encoded representation of the following text.

From:user@example.com
Subject: Test
Message sent using SendRawEmail.
```

The following is a complete POST request to the SendRawEmail operator. The request includes the X-Amzn-Authorization header. None of the headers are URL-encoded.

```
POST / HTTP/1.1
Host: email.us-west-2.amazonaws.com
Content-Type: application/x-www-form-urlencoded
Date: Tue, 25 May 2010 21:20:27 +0000
Content-Length: 174

Authorization: AWS4-HMAC-SHA256 Credential=AKIAXEXAMPLE/20150830/us-east-1/iam/aws4_request,
  SignedHeaders=content-type;host;x-amz-date,Signature=5d672d79c1EXAMPLE

Action=SendRawEmail
&Destinations.member.1=allan%40example.com
&RawMessage.Data=RnJvbTp1c2VyQGV4YW1wbGUuY29tDQpTdWJqZWN0OiBUZXN0DQoNCk1lc3 ...
```

Amazon SES API responses

In response to an API request, Amazon SES returns an XML data structure that contains the results of the request.

Every Amazon SES response includes a request ID in a RequestId element. The value is a unique string that AWS assigns. If you ever have issues with a particular request, AWS will ask for the request ID to help troubleshoot the issue.

Successful Amazon SES responses also include one or more message IDs. You can think of a message ID as a receipt for an email message that Amazon SES sends. If a message is rejected or bounced, the
message ID will appear in any complaint or bounce notifications that you receive; you can then use the message ID to identify any problematic email messages that you have sent, and take corrective action.

**Structure of a successful response**

If the request succeeded, the main response element is named after the action, but with “Response” appended. For example, **SendEmailResponse** is the response element returned for a successful **SendEmail** request. This element contains the following child elements:

- **ResponseMetadata**, which contains the **RequestId** child element.
- An optional element containing action-specific results. For example, the **SendEmailResponse** element includes an element called **SendEmailResult**.

The XML schema describes the XML response message for each Amazon SES action.

The following is an example of a successful response.

```
<SendEmailResponse xmlns="https://email.amazonaws.com/doc/2010-03-31/">
  <SendEmailResult>
    <MessageId>000001271b15238a-fd3ae762-2563-11df-8cd4-6d4e828a9ae8-000000</MessageId>
  </SendEmailResult>
  <ResponseMetadata>
    <RequestId>fd3ae762-2563-11df-8cd4-6d4e828a9ae8</RequestId>
  </ResponseMetadata>
</SendEmailResponse>
```

**Structure of an Amazon SES API error response**

If a request is unsuccessful, the main response element is called **ErrorResponse** regardless of the action that was called. This element contains an **Error** element and a **RequestId** element. Each **Error** includes:

- A **Type** element that identifies whether the error was a receiver or sender error
- A **Code** element that identifies the type of error that occurred
- A **Message** element that describes the error condition in a human-readable form
- A **Detail** element that might give additional details about the error or might be empty

The following is an example of an error response.

```
<ErrorResponse>
  <Error>
    <Type>Sender</Type>
    <Code>Validation</Code>
    <Message>
      Value null at 'message.subject' failed to satisfy constraint: Member must not be null
    </Message>
  </Error>
  <RequestId>42d59b56-7407-4c4a-be0f-4c88daeea257</RequestId>
</ErrorResponse>
```
Error codes returned by the Amazon SES API

This topic contains a list of error codes that are returned by the Amazon SES API. For more information about the Amazon SES API, see the Amazon Simple Email Service API Reference.

You should retry HTTPS requests that receive 5xx errors. In this case, to reduce the likelihood of generating duplicates, we recommend that you implement an exponential retry method with progressively longer waits (5, 10, and 30 seconds) between consecutive timeouts. If the third retry call does not succeed, perform another set of retries after 20 minutes. For an example implementation that uses an exponential retry policy with Amazon SES, see How to handle a "Throttling - Maximum sending rate exceeded" error on the Amazon SES blog.

**Note**

AWS SDKs implement retry logic automatically.

HTTPS client errors (4xx) indicate that you need to revise the request to correct the problem before trying again. For example, if your AWS authentication credentials are invalid, you must update your setup to use the proper credentials before trying to send the email again.

<table>
<thead>
<tr>
<th>Error</th>
<th>Description</th>
<th>HTTPS Status Code</th>
<th>Actions That Return This Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConfigurationSetDoesNotExist</td>
<td>The specified configuration set does not exist. A configuration set is an optional parameter that you use to publish email sending events. For more information, see Monitor email sending using Amazon SES event publishing (p. 288).</td>
<td>400</td>
<td>SendEmail, SendRawEmail</td>
</tr>
<tr>
<td>IncompleteSignature</td>
<td>The request signature does not conform to AWS standards.</td>
<td>400</td>
<td>All</td>
</tr>
<tr>
<td>InternalFailure</td>
<td>The request processing has failed because of an unknown error, exception, or failure.</td>
<td>500</td>
<td>All</td>
</tr>
<tr>
<td>InvalidAction</td>
<td>The requested action or operation is invalid. Verify that the action is typed correctly.</td>
<td>400</td>
<td>All</td>
</tr>
<tr>
<td>InvalidClientTokenId</td>
<td>The X.509 certificate or AWS access key ID provided does not exist in our records.</td>
<td>403</td>
<td>All</td>
</tr>
<tr>
<td>InvalidParameterCombination</td>
<td>Parameters that must not be used together were used together.</td>
<td>400</td>
<td>All</td>
</tr>
<tr>
<td>Error</td>
<td>Description</td>
<td>HTTPS Status Code</td>
<td>Actions That Return This Code</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td>InvalidParameterValue</td>
<td>An invalid or out-of-range value was supplied for the input parameter.</td>
<td>400</td>
<td>All</td>
</tr>
<tr>
<td>InvalidQueryParameter</td>
<td>The AWS query string is malformed, does not adhere to AWS standards.</td>
<td>400</td>
<td>All</td>
</tr>
<tr>
<td>MailFromDomainNotVerified</td>
<td>The message could not be sent because Amazon SES could not read the MX record required to use the specified MAIL FROM domain.</td>
<td>400</td>
<td>SendEmail, SendRawEmail</td>
</tr>
<tr>
<td>MalformedQueryString</td>
<td>The query string contains a syntax error.</td>
<td>404</td>
<td>All</td>
</tr>
<tr>
<td>MessageRejected</td>
<td>Indicates that the action failed, and the message could not be sent. Check the error stack for a description of what caused the error. For more information about problems that can cause this error, see Amazon SES email sending errors (p. 476).</td>
<td>400</td>
<td>SendEmail, SendRawEmail</td>
</tr>
<tr>
<td>MissingAction</td>
<td>The request is missing an action or a required parameter.</td>
<td>400</td>
<td>All</td>
</tr>
<tr>
<td>MissingAuthenticationToken</td>
<td>The request must contain either a valid (registered) AWS access key ID or X.509 certificate.</td>
<td>403</td>
<td>All</td>
</tr>
<tr>
<td>MissingParameter</td>
<td>A required parameter for the specified action is not supplied.</td>
<td>400</td>
<td>All</td>
</tr>
<tr>
<td>OptInRequired</td>
<td>The AWS access key ID needs a subscription for the service.</td>
<td>403</td>
<td>All</td>
</tr>
<tr>
<td>Error</td>
<td>Description</td>
<td>HTTPS Status Code</td>
<td>Actions That Return This Code</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------------------------------------------------------------------</td>
<td>-------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>RequestExpired</td>
<td>The request reached the service more than 15 minutes after the date stamp on the request or more than 15 minutes after the request expiration date (such as for pre-signed URLs), or the date stamp on the request is more than 15 minutes in the future.</td>
<td>400</td>
<td>All</td>
</tr>
<tr>
<td>ServiceUnavailable</td>
<td>The request failed due to a temporary failure of the server.</td>
<td>503</td>
<td>All</td>
</tr>
<tr>
<td>Throttling</td>
<td>The request was denied due to request throttling.</td>
<td>400</td>
<td>All</td>
</tr>
</tbody>
</table>
Amazon SES code examples

This section contains code examples that help you get started using Amazon SES to send email and manage your Amazon SES account. Code examples are available in the following languages: C#, Go, Java, PHP, Python, and Ruby.

Choose one of the following links to see code examples for that task:
- Sending email using AWS SDKs (p. 416)
- Sending email using the Amazon SES SMTP Interface (p. 429)
- Sending raw email using AWS SDKs (p. 442)
- Verify multiple email addresses (p. 450)
- Replicating email identities across AWS Regions (p. 451)

Sending email using AWS SDKs

The AWS SDKs contain built-in methods for interacting with Amazon SES and several other AWS services. If you plan to use Amazon SES along with other AWS services, we recommend that you use an SDK. To learn more about the AWS SDKs, see Tools for Amazon Web Services.

In this section, you will find code examples in several programming languages that demonstrate the process of sending email through Amazon SES using the AWS SDKs.

C#

The following code example is a complete solution for sending email through Amazon SES using the AWS SDK for .NET. This code example assumes that you have installed the AWS SDK for .NET, and that you’ve created a shared credentials file. For more information about creating a shared credentials file, see Create a shared credentials file (p. 31).

Important
You use a shared credentials file to pass your AWS access key ID and secret access key. As an alternative to using a shared credentials file, you can specify your AWS access key ID and secret access key in the SDK Store. For more information, see Configuring AWS credentials in the AWS SDK for .NET Developer Guide. This example doesn’t function unless you specify your credentials using one of these methods.

```csharp
using Amazon;
using System;
using System.Collections.Generic;
using Amazon.SimpleEmail;
using Amazon.SimpleEmail.Model;

namespace AmazonSESSample
{
    class Program
    {
        // Replace sender@example.com with your "From" address.
        // This address must be verified with Amazon SES.
        static readonly string senderAddress = "sender@example.com";

        // Replace recipient@example.com with a "To" address. If your account
        // is still in the sandbox, this address must be verified.
        static readonly string receiverAddress = "recipient@example.com";
```
// The configuration set to use for this email. If you do not want to use a
// configuration set, comment out the following property and the
// ConfigurationSetName = configSet argument below.
static readonly string configSet = "ConfigSet";

// The subject line for the email.
static readonly string subject = "Amazon SES test (AWS SDK for .NET)";

// The email body for recipients with non-HTML email clients.
static readonly string textBody = "Amazon SES Test (.NET)
+ "This email was sent through Amazon SES "
+ "using the AWS SDK for .NET."

// The HTML body of the email.
static readonly string htmlBody = @"<html>
<head></head>
<body>
<h1>Amazon SES Test (AWS SDK for .NET)</h1>
This email was sent with
<a href='https://aws.amazon.com/ses/'>Amazon SES</a> using the
<a href='https://aws.amazon.com/sdk-for-net/'>AWS SDK for .NET</a>.</body>
</html>";

static void Main(string[] args)
{
    // Replace USWest2 with the AWS Region you're using for Amazon SES.
    // Acceptable values are EUWest1, USEast1, and USWest2.
    using (var client = new AmazonSimpleEmailServiceClient(RegionEndpoint.USWest2))
    {
        var sendRequest = new SendEmailRequest
        {
            Source = senderAddress,
            Destination = new Destination
            {
                ToAddresses =
                new List<string> { receiverAddress }
            },
            Message = new Message
            {
                Subject = new Content(subject),
                Body = new Body
                {
                    Html = new Content
                    {
                        Charset = "UTF-8",
                        Data = htmlBody
                    },
                    Text = new Content
                    {
                        Charset = "UTF-8",
                        Data = textBody
                    }
                }
            },
            // If you are not using a configuration set, comment
            // or remove the following line
            ConfigurationSetName = configSet
        };
        try
        {
            Console.WriteLine("Sending email using Amazon SES... ");
            var response = client.SendEmail(sendRequest);
            Console.WriteLine("The email was sent successfully.");
        }
        catch
        {
            Console.WriteLine("Failed to send email.");
        }
    }
}
The following code example is a complete solution for sending email through Amazon SES using the
AWS SDK for Go. This code example assumes that you have installed the AWS SDK for Go, and that
you have created a shared credentials file. For more information about creating a shared credentials
file, see Create a shared credentials file (p. 31).

**Important**

You use a shared credentials file to pass your AWS access key ID and secret access key. As
an alternative to using a shared credentials file, you can specify your AWS access key ID
and secret access key by setting two environment variables (AWS_ACCESS_KEY_ID and
AWS_SECRET_ACCESS_KEY, respectively). This example doesn't function unless you specify
your credentials using one of these methods.

```go
package main

import (
    "fmt"
    //go get -u github.com/aws/aws-sdk-go
    "github.com/aws/aws-sdk-go/aws"
    "github.com/aws/aws-sdk-go/aws/session"
    "github.com/aws/aws-sdk-go/service/ses"
    "github.com/aws/aws-sdk-go/aws/awserr"
)

const (
    // Replace sender@example.com with your "From" address.
    // This address must be verified with Amazon SES.
    Sender = "sender@example.com"

    // Replace recipient@example.com with a "To" address. If your account
    // is still in the sandbox, this address must be verified.
    Recipient = "recipient@example.com"

    // Specify a configuration set. If you do not want to use a configuration
    // set, comment out the following constant and the
    // ConfigurationSetName: aws.String(ConfigurationSet) argument below
    ConfigurationSet = "ConfigSet"

    // Replace us-west-2 with the AWS Region you're using for Amazon SES.
    AwsRegion = "us-west-2"

    // The subject line for the email.
    Subject = "Amazon SES Test (AWS SDK for Go)"

    // The HTML body for the email.
    HtmlBody = "<h1>Amazon SES Test Email (AWS SDK for Go)</h1><p>This email was sent with " +
```
"<a href='https://aws.amazon.com/ses/'>Amazon SES</a> using the " + "<a href='https://aws.amazon.com/sdk-for-go/'>AWS SDK for Go</a>.</p>"

//The email body for recipients with non-HTML email clients.
TextBody = "This email was sent with Amazon SES using the AWS SDK for Go."

// The character encoding for the email.
CharSet = "UTF-8"

func main() {
    // Create a new session and specify an AWS Region.
sess, err := session.NewSession(&aws.Config{
        Region: aws.String(AwsRegion),
    })
    if err != nil {
        return
    }

    // Create an SES client in the session.
svc := ses.New(sess)

    // Assemble the email.
    input := &ses.SendEmailInput{
        Destination: &ses.Destination{
            ToAddresses: []*string{aws.String(Recipient),},
        },
        Message: &ses.Message{
            Body: &ses.Body{
                Html: &ses.Content{
                    Charset: aws.String(CharSet),
                    Data:    aws.String(HtmlBody),
                },
                Text: &ses.Content{
                    Charset: aws.String(CharSet),
                    Data:    aws.String(TextBody),
                },
            },
            Subject: &ses.Content{
                Charset: aws.String(CharSet),
                Data:    aws.String(Subject),
            },
        },
        Source: aws.String(Sender),
    }

    // Attempt to send the email.
    result, err := svc.SendEmail(input)
    if err != nil {
        if aerr, ok := err.(awserr.Error); ok {
            switch aerr.Code() {
            case ses.ErrCodeMessageRejected:
            case ses.ErrCodeMailFromDomainNotVerifiedException:
            case ses.ErrCodeConfigurationSetNameDoesNotExistException:
                fmt.Println(ses.ErrCodeConfigurationSetNameDoesNotExistException, aerr.Error())
            }
        }
    }
}
Java SDK v1

The following code example is a complete solution for sending email through Amazon SES using the AWS SDK for Java. This code example assumes that you have installed the AWS SDK for Java, and that you have created a shared credentials file. For more information about creating a shared credentials file, see Create a shared credentials file (p. 31).

```java
package com.amazonaws.samples;

import java.io.IOException;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.simpleemail.AmazonSimpleEmailService;
import com.amazonaws.services.simpleemail.AmazonSimpleEmailServiceClientBuilder;
import com.amazonaws.services.simpleemail.model.Body;
import com.amazonaws.services.simpleemail.model.Content;
import com.amazonaws.services.simpleemail.model.Destination;
import com.amazonaws.services.simpleemail.model.Message;
import com.amazonaws.services.simpleemail.model.SendEmailRequest;

public class AmazonSESSample {

    // Replace sender@example.com with your "From" address.
    // This address must be verified with Amazon SES.
    static final String FROM = "sender@example.com";

    // Replace recipient@example.com with a "To" address. If your account
    // is still in the sandbox, this address must be verified.
    static final String TO = "recipient@example.com";

    // The configuration set to use for this email. If you do not want to use a
    // configuration set, comment the following variable and the
    // .withConfigurationSetName(CONFIGSET); argument below.
    static final String CONFIGSET = "ConfigSet";

    // The subject line for the email.
    static final String SUBJECT = "Amazon SES test (AWS SDK for Java)";

    // The HTML body for the email.
    static final String HTMLBODY = "<h1>Amazon SES test (AWS SDK for Java)</h1>"
        + "<p>This email was sent with <a href='https://aws.amazon.com/ses/'>Amazon SES</a> using the <a href='https://aws.amazon.com/sdk-for-java/'>AWS SDK for Java</a></p>";

    // The email body for recipients with non-HTML email clients.
    static final String TEXTBODY = "This email was sent through Amazon SES using the AWS SDK for Java."

    public static void main(String[] args) throws IOException {
        System.out.println("Email Sent!");
        System.out.println(result);
    }
}
```
try {
    AmazonSimpleEmailService client =
    AmazonSimpleEmailServiceClientBuilder.standard()
     // Replace US_WEST_2 with the AWS Region you're using for
     // Amazon SES.
     .withRegion(Regions.US_WEST_2).build();
    SendEmailRequest request = new SendEmailRequest()
     .withDestination(  
      new Destination().withToAddresses(TO))
     .withMessage(new Message()  
       .withBody(new Body()  
         .withHtml(new Content()  
           .withCharset("UTF-8").withData(HTMLBODY))  
         .withText(new Content()  
           .withCharset("UTF-8").withData(TEXTBODY))
         .withSubject(new Content()  
           .withCharset("UTF-8").withData(SUBJECT)))
       .withSource(FROM)
     // Comment or remove the next line if you are not using a
     // configuration set
     .withConfigurationSetName(CONFIGSET);
    client.sendEmail(request);
    System.out.println("Email sent!");
} catch (Exception ex) {
    System.out.println("The email was not sent. Error message: " + ex.getMessage());
}

Java SDK v2

The following code example is a complete solution for sending email through Amazon SES using the
AWS SDK for Java 2.x and the JavaMail API. This code example assumes that you have installed the
SDK for Java 2.x, and that you have created a shared credentials file. For more information about
creating a shared credentials file, see Create a shared credentials file (p. 31).

```java
package com.example.ses;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.ses.SesClient;
import javax.mail.Message;
import javax.mail.MessagingException;
import javax.mail.Session;
import javax.mail.internet.AddressException;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;
import javax.mail.internet.MimeMultipart;
import javax.mail.internet.MimeBodyPart;
import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.nio.ByteBuffer;
import java.util.Properties;
import software.amazon.awssdk.core.SdkBytes;
import software.amazon.awssdk.services.ses.model.SendRawEmailRequest;
import software.amazon.awssdk.services.ses.model.RawMessage;
import software.amazon.awssdk.services.ses.model.SesException;

/**
 * To run this Java V2 code example, ensure that you have setup your development
 * environment, including your credentials.
 *
 * For information, see this documentation topic:
 */
public class SendMessage {

    public static void main(String[] args) throws IOException {

        final String USAGE = "\n" + "Usage:\n" + "SendMessage <sender> <recipient> <subject> \n\n" + "Where:\n" + " sender - an email address that represents the sender. \n" + " recipient - an email address that represents the recipient. \n" + " subject - the subject line. \n";

        if (args.length != 3) {
            System.out.println(USAGE);
            System.exit(1);
        }

        String sender = args[0];
        String recipient = args[1];
        String subject = args[2];

        Region region = Region.US_EAST_1;
        SesClient client = SesClient.builder()
            .region(region)
            .build();

        // The email body for non-HTML email clients
        String bodyText = "Hello,\n\r\nSee the list of customers. \n";

        // The HTML body of the email
        String bodyHTML = "<html>" + "<head></head>" + "<body>" + "<h1>Hello!</h1>" + "<p>See the list of customers.</p>" + "</body>" + "</html>";

        try {
            send(client, sender, recipient, subject, bodyText, bodyHTML);
            client.close();
            System.out.println("Done");
        } catch (IOException | MessagingException e) {
            e.printStackTrace();
        }
    }

    public static void send(SesClient client,
            String sender, String recipient, String subject,
            String bodyText, String bodyHTML)
            throws AddressException, MessagingException, IOException {

        Session session = Session.getDefaultInstance(new Properties());
        MimeMessage message = new MimeMessage(session);

        // Add subject, from and to lines
        message.setSubject(subject, "UTF-8");
        message.setFrom(new InternetAddress(sender));
        message.setRecipients(Message.RecipientType.TO, InternetAddress.parse(recipient));

        // Create a multipart/alternative child container
        MimeMultipart msgBody = new MimeMultipart("alternative");

        Multipart part = msgBody.addPart(bodyText);
        MimeBodyPart htmlPart = new MimeBodyPart(bodyHTML);

        // Add the HTML content
        htmlPart.setHeader("Content-Type", "text/html; charset=UTF-8");
        htmlPart.setText(bodyHTML);

        part.addPart(htmlPart);
    }
}
// Create a wrapper for the HTML and text parts
MimeBodyPart wrap = new MimeBodyPart();

// Define the text part
MimeBodyPart textPart = new MimeBodyPart();
textPart.setContent(bodyText, "text/plain; charset=UTF-8");

// Define the HTML part
MimeBodyPart htmlPart = new MimeBodyPart();
htmlPart.setContent(bodyHTML, "text/html; charset=UTF-8");

// Add the text and HTML parts to the child container
msgBody.addBodyPart(textPart);
msgBody.addBodyPart(htmlPart);

// Add the child container to the wrapper object
wrap.setContent(msgBody);

// Create a multipart/mixed parent container
MimeMultipart msg = new MimeMultipart("mixed");

// Add the parent container to the message
message.setContent(msg);

// Add the multipart/alternative part to the message
msg.addBodyPart(wrap);

try {
    System.out.println("Attempting to send an email through Amazon SES using the AWS SDK for Java...");

    ByteArrayOutputStream outputStream = new ByteArrayOutputStream();
    message.writeTo(outputStream);
    ByteBuffer buf = ByteBuffer.wrap(outputStream.toByteArray());
    byte[] arr = new byte[buf.remaining()];
    buf.get(arr);
    SdkBytes data = SdkBytes.fromByteArray(arr);
    RawMessage rawMessage = RawMessage.builder()
        .data(data)
        .build();
    SendRawEmailRequest rawEmailRequest = SendRawEmailRequest.builder()
        .rawMessage(rawMessage)
        .build();
    client.sendRawEmail(rawEmailRequest);
} catch (SesException e) {
    System.err.println(e.awsErrorDetails().errorMessage());
    System.exit(1);
}


JavaScript

The following code example is a complete solution for sending email through Amazon SES using the AWS SDK for JavaScript in Node.js. This code example assumes that you have installed the SDK for JavaScript in Node.js. You must also create a configuration file that contains your AWS Access Key
ID, Secret Access Key, and preferred AWS Region. For more information about creating this file, see Loading Credentials in Node.js from a JSON File.

**Important**

You use a shared credentials file to pass your AWS access key ID and secret access key. As an alternative to using a shared credentials file, you can specify your AWS access key ID and secret access key by setting two environment variables (`AWS_ACCESS_KEY_ID` and `AWS_SECRET_ACCESS_KEY`, respectively). This example doesn't function unless you specify your credentials using one of these methods.

```javascript
'use strict';

var aws = require('aws-sdk');

// Provide the full path to your config.json file.
aws.config.loadFromPath('./config.json');

// Replace sender@example.com with your "From" address.
// This address must be verified with Amazon SES.
const sender = "Sender Name <sender@example.com>";

// Replace recipient@example.com with a "To" address. If your account
// is still in the sandbox, this address must be verified.
const recipient = "recipient@example.com";

// Specify a configuration set. If you do not want to use a configuration
// set, comment the following variable, and the
// ConfigurationSetName : configuration_set argument below.
const configuration_set = "ConfigSet";

// The subject line for the email.
const subject = "Amazon SES Test (AWS SDK for JavaScript in Node.js)";

// The email body for recipients with non-HTML email clients.
const body_text = "Amazon SES Test (SDK for JavaScript in Node.js)\r\n" + "This email was sent with Amazon SES using the " + "AWS SDK for JavaScript in Node.js.";

// The HTML body of the email.
const body_html = `<html>
<head></head>
<body>
<h1>Amazon SES Test (SDK for JavaScript in Node.js)</h1>
<p>This email was sent with
   <a href='https://aws.amazon.com/ses/'>Amazon SES</a> using the
   <a href='https://aws.amazon.com/sdk-for-node-js/'>AWS SDK for JavaScript in Node.js</a>.</p>
</body>
</html>`;

// The character encoding for the email.
const charset = "UTF-8";

// Create a new SES object.
var ses = new aws.SES();

// Specify the parameters to pass to the API.
var params = {
  Source: sender,
  Destination: {
    ToAddresses: [recipient]
  },
  Message: {
    Subject: {
      Data: subject,
      Charset: charset
    },
    Body: {
      Text: {
        Data: body_text,
        Charset: charset
      },
      Html: {
        Data: body_html,
        Charset: charset
      }
    }
  }
};

// Send the email.
var promise = ses.sendEmail(params, function(err, data) { if (err) { console.log(err, err.stack); } else { console.log("Email sent!"醢储备数据.); } });
```

424
Subject: {
    Data: subject,
    Charset: charset
},
Body: {
    Text: {
        Data: body_text,
        Charset: charset
    },
    Html: {
        Data: body_html,
        Charset: charset
    }
},
ConfigurationSetName: configuration_set
};

// Try to send the email.
ses.sendEmail(params, function(err, data) {
    // If something goes wrong, print an error message.
    if(err) {
        console.log(err.message);
    } else {
        console.log("Email sent! Message ID: ", data.MessageId);
    }
});

---

**PHP**

The following code example is a complete solution for sending email through Amazon SES using the AWS SDK for PHP. This code example assumes that you have installed the AWS SDK for PHP, and that you have created a shared credentials file. For more information about creating a shared credentials file, see Create a shared credentials file (p. 31).

**Important**

You use a shared credentials file to pass your AWS access key ID and secret access key. As an alternative to using a shared credentials file, you can specify your AWS access key ID and secret access key by setting two environment variables (`AWS_ACCESS_KEY_ID` and `AWS_SECRET_ACCESS_KEY`, respectively). This example doesn't function unless you specify your credentials using one of these methods.

```php
<?php

// If necessary, modify the path in the require statement below to refer to the
// location of your Composer autoload.php file.
require 'vendor/autoload.php';

use Aws\Ses\SesClient;
use Aws\Exception\AwsException;

// Create an SesClient. Change the value of the region parameter if you're
// using an AWS Region other than US West (Oregon). Change the value of the
// profile parameter if you want to use a profile in your credentials file
// other than the default.
$SesClient = new SesClient([
    'profile' => 'default',
    'version' => '2010-12-01',
    'region'  => 'us-west-2'
]);

// Replace sender@example.com with your "From" address.
// This address must be verified with Amazon SES.
$search_email = 'sender@example.com';
```
// Replace these sample addresses with the addresses of your recipients. If
// your account is still in the sandbox, these addresses must be verified.
$recipient_emails = ['recipient1@example.com', 'recipient2@example.com'];

// Specify a configuration set. If you do not want to use a configuration
// set, comment the following variable, and the
// 'ConfigurationSetName' => $configuration_set argument below.
$configuration_set = 'ConfigSet';

$subject = 'Amazon SES test (AWS SDK for PHP)';
$plaintext_body = 'This email was sent with Amazon SES using the AWS SDK for PHP.' ;
$html_body = '<h1>AWS Amazon Simple Email Service Test Email</h1>'.
'<p>This email was sent with a href="https://aws.amazon.com/ses/">Amazon SES</a> using the a href="https://aws.amazon.com/sdk-for-
php/">AWS SDK for PHP</a></p>.

$char_set = 'UTF-8';

try {
    $result = $SesClient->sendEmail([
    'Destination' => [
        'ToAddresses' => $recipient_emails,
    ],
    'ReplyToAddresses' => [$sender_email],
    'Source' => $sender_email,
    'Message' => [
        'Body' => [
            'Html' => [
                'Charset' => $char_set,
                'Data' => $html_body,
            ],
            'Text' => [
                'Charset' => $char_set,
                'Data' => $plaintext_body,
            ],
        ],
        'Subject' => [
            'Charset' => $char_set,
            'Data' => $subject,
        ],
    ],
    // If you aren't using a configuration set, comment or delete the
    // following line
    'ConfigurationSetName' => $configuration_set,
    ]);
    $messageId = $result['MessageId'];
    echo("Email sent! Message ID: $messageId\n");
} catch (AwsException $e) {
    // output error message if fails
    echo $e->getMessage();
    echo("The email was not sent. Error message: ".$e->getAwsErrorMessage()."\n");
    echo "\n";
}

Python

The following code example is a complete solution for sending email through Amazon SES using
the AWS SDK for Python (Boto). This code example assumes that you have installed the AWS SDK
for Python (Boto), and that you have created a shared credentials file. For more information about
creating a shared credentials file, see Create a shared credentials file (p. 31).

Important
You use a shared credentials file to pass your AWS access key ID and secret access key. As
an alternative to using a shared credentials file, you can specify your AWS access key ID
and secret access key by setting two environment variables (AWS_ACCESS_KEY_ID and AWS_SECRET_ACCESS_KEY, respectively). This example doesn't function unless you specify your credentials using one of these methods.

```python
import boto3
from botocore.exceptions import ClientError

# Replace sender@example.com with your "From" address.
# This address must be verified with Amazon SES.
SENDER = "Sender Name <sender@example.com>"

# Replace recipient@example.com with a "To" address. If your account
# is still in the sandbox, this address must be verified.
RECIPIENT = "recipient@example.com"

# Specify a configuration set. If you do not want to use a configuration
# set, comment the following variable, and the
# ConfigurationSetName=CONFIGURATION_SET argument below.
CONFIGURATION_SET = "ConfigSet"

# If necessary, replace us-west-2 with the AWS Region you're using for Amazon SES.
AWS_REGION = "us-west-2"

# The subject line for the email.
SUBJECT = "Amazon SES Test (SDK for Python)"

# The email body for recipients with non-HTML email clients.
BODY_TEXT = ("Amazon SES Test (Python)\r\n" "This email was sent with Amazon SES using the " "AWS SDK for Python (Boto)."
}

# The HTML body of the email.
BODY_HTML = """
<html>
<head></head>
<body>
<h1>Amazon SES Test (SDK for Python)</h1>
<p>This email was sent with
<a href='https://aws.amazon.com/ses/'>Amazon SES</a> using the
<a href='https://aws.amazon.com/sdk-for-python/'>AWS SDK for Python (Boto)</a>.</p>
</body>
</html>
"

# The character encoding for the email.
CHARSET = "UTF-8"

# Create a new SES resource and specify a region.
client = boto3.client('ses',region_name=AWS_REGION)

# Try to send the email.
try:
    #Provide the contents of the email.
    response = client.send_email(
        Destination=
        {
            'ToAddresses': [RECIPIENT],
        },
        Message={
            'Body': {
                'Html': {
                    'Charset': CHARSET,
                    'Data': BODY_HTML,
                },
            },
        },
    )
```
The following code example is a complete solution for sending email through Amazon SES using the AWS SDK for Ruby. This code example assumes that you've installed the AWS SDK for Ruby, and that you've created a shared credentials file. For more information about installing the SDK for Ruby, see Installing the AWS SDK for Ruby in the AWS SDK for Ruby Developer Guide. For more information about creating a shared credentials file, see Create a shared credentials file (p. 31).

**Important**
You use a shared credentials file to pass your AWS access key ID and secret access key. As an alternative to using a shared credentials file, you can specify your AWS access key ID and secret access key by setting two environment variables (`AWS_ACCESS_KEY_ID` and `AWS_SECRET_ACCESS_KEY`, respectively). This example doesn't function unless you specify your credentials using one of these methods.

```ruby
require 'aws-sdk'

# Replace sender@example.com with your "From" address.
# This address must be verified with Amazon SES.
sender = "sender@example.com"

# Replace recipient@example.com with a "To" address. If your account
# is still in the sandbox, this address must be verified.
recipient = "recipient@example.com"

# Specify a configuration set. If you do not want to use a configuration
# set, comment the following variable and the
# configuration_set_name: configsetname argument below.
configsetname = "ConfigSet"

# Replace us-west-2 with the AWS Region you’re using for Amazon SES.
awsregion = "us-west-2"

# The subject line for the email.
sender = "Amazon SES test (AWS SDK for Ruby)"

# The HTML body of the email.
htmlbody = "<h1>Amazon SES test (AWS SDK for Ruby)<h1>"
# If you are not using a configuration set, comment or delete the
# following line
ConfigurationSetName=CONFIGURATION_SET,
except ClientError as e:
    print(e.response['Error']['Message'])
else:
    print("Email sent! Message ID:"),
    print(response['MessageId'])
```
Sending email using the Amazon SES SMTP Interface

Several programming languages include standard libraries for sending email using SMTP. You can use these libraries to create email sending applications that are lightweight and highly configurable.

In this section, you will find code examples in several programming languages that demonstrate the process of sending email through Amazon SES using the SMTP interface. Wherever possible, these code examples use standard libraries.
C#

The following code example is a complete solution for sending email through the Amazon SES SMTP interface using C#. In order to run this code example, you must obtain SMTP credentials; for more information, see Obtaining your Amazon SES SMTP credentials (p. 86).

```csharp
using System;
using System.Net;
using System.Net.Mail;

namespace AmazonSESSample
{
    class Program
    {
        static void Main(string[] args)
        {
            // Replace sender@example.com with your "From" address.
            // This address must be verified with Amazon SES.
            String FROM = "sender@example.com";
            String FROMNAME = "Sender Name";

            // Replace recipient@example.com with a "To" address. If your account
            // is still in the sandbox, this address must be verified.
            String TO = "recipient@amazon.com";

            // Replace smtp_username with your Amazon SES SMTP user name.
            String SMTP_USERNAME = "smtp_username";

            // Replace smtp_password with your Amazon SES SMTP password.
            String SMTP_PASSWORD = "smtp_password";

            // (Optional) the name of a configuration set to use for this message.
            // If you comment out this line, you also need to remove or comment out
            // the "X-SES-CONFIGURATION-SET" header below.
            String CONFIGSET = "ConfigSet";

            // If you're using Amazon SES in a region other than US West (Oregon),
            // replace email-smtp.us-west-2.amazonaws.com with the Amazon SES SMTP
            // endpoint in the appropriate AWS Region.
            String HOST = "email-smtp.us-west-2.amazonaws.com";

            // The port you will connect to on the Amazon SES SMTP endpoint. We
            // are choosing port 587 because we will use STARTTLS to encrypt
            // the connection.
            int PORT = 587;

            // The subject line of the email
            String SUBJECT = "Amazon SES test (SMTP interface accessed using C#)";

            // The body of the email
            String BODY = "<h1>Amazon SES Test</h1>" +
                "<p>This email was sent through the " +
                "<a href='https://aws.amazon.com/ses'>Amazon SES</a> SMTP interface " +
                "using the .NET System.Net.Mail library.</p>";

            // Create and build a new MailMessage object
            MailMessage message = new MailMessage();
            message.IsBodyHtml = true;
            message.From = new MailAddress(FROM, FROMNAME);
            message.To.Add(new MailAddress(TO));
            message.Subject = SUBJECT;
            message.Body = BODY;
        }
    }
}
```

430
message.Headers.Add("X-SES-CONFIGURATION-SET", CONFIGSET);

{
    // Pass SMTP credentials
    client.Credentials =
        new NetworkCredential(SMTP_USERNAME, SMTP_PASSWORD);

    // Enable SSL encryption
    client.EnableSsl = true;

    // Try to send the message. Show status in console.
    try
    {
        Console.WriteLine("Attempting to send email...");
        client.Send(message);
        Console.WriteLine("Email sent!");
    }
    catch (Exception ex)
    {
        Console.WriteLine("The email was not sent.");
        Console.WriteLine("Error message: " + ex.Message);
    }
}
}
}

Go

The following code example is a complete solution for sending email through the Amazon SES SMTP interface using the Go programming language. In order to run this code example, you must obtain SMTP credentials; for more information, see Obtaining your Amazon SES SMTP credentials (p. 86). You must also install the Gomail package.

```go
package main

import (   "fmt"   "gopkg.in/gomail.v2" // go get gopkg.in/gomail.v2 )

const (   // Replace sender@example.com with your "From" address.   // This address must be verified with Amazon SES.   Sender = "sender@example.com"   SenderName = "Sender Name"

   // Replace recipient@example.com with a "To" address. If your account   // is still in the sandbox, this address must be verified.   Recipient = "recipient@example.com"

   // Replace SmtpUser with your Amazon SES SMTP user name.   SmtpUser = "SmtpUser"

   // Replace SmtpPass with your Amazon SES SMTP password.   SmtpPass = "SmtpPass"

   // The name of the configuration set to use for this message.   // If you comment out or remove this variable, you will also need to   // comment out or remove the header below.   ConfigSet = "ConfigSet"

   // If you're using Amazon SES in an AWS Region other than US West (Oregon),
```
func main() {

  // Create a new message.
  m := gomail.NewMessage()

  // Set the main email part to use HTML.
  m.SetBody("text/html", HtmlBody)

  // Set the alternative part to plain text.
  m.AddAlternative("text/plain", TextBody)

  // Construct the message headers, including a Configuration Set and a Tag.
  m.SetHeaders(map[string][]string{
      "From": {m.FormatAddress(Sender, SenderName)},
      "To": {Recipient},
      "Subject": {Subject},
      // Comment or remove the next line if you are not using a configuration set
      "X-SES-CONFIGURATION-SET": {ConfigSet},
      // Comment or remove the next line if you are not using custom tags
      "X-SES-MESSAGE-TAGS": {Tags},
  })

  // Send the email.
  d := gomail.NewPlainDialer(Host, Port, SmtpUser, SmtpPass)

  // Display an error message if something goes wrong; otherwise,
  // display a message confirming that the message was sent.
  if err := d.DialAndSend(m); err != nil {
    fmt.Println(err)
  } else {
    fmt.Println("Email sent!")
  }
}
Java

The following code example is a complete solution for sending email through the Amazon SES SMTP interface using Java. In order to run this code example, you must obtain SMTP credentials; for more information, see Obtaining your Amazon SES SMTP credentials (p. 86). You must also download the JavaMail API.

```java
import java.util.Properties;
import javax.mail.Message;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;

public class AmazonSESSample {
    // Replace sender@example.com with your "From" address.
    // This address must be verified.
    static final String FROM = "sender@example.com";
    static final String FROMNAME = "Sender Name";

    // Replace recipient@example.com with a "To" address. If your account
    // is still in the sandbox, this address must be verified.
    static final String TO = "recipient@example.com";

    // Replace smtp_username with your Amazon SES SMTP user name.
    static final String SMTP_USERNAME = "smtp_username";

    // Replace smtp_password with your Amazon SES SMTP password.
    static final String SMTP_PASSWORD = "smtp_password";

    // The name of the Configuration Set to use for this message.
    // If you comment out or remove this variable, you will also need to
    // comment out or remove the header below.
    static final String CONFIGSET = "ConfigSet";

    // Amazon SES SMTP host name. This example uses the US West (Oregon) region.
    // See https://docs.aws.amazon.com/ses/latest/DeveloperGuide/regions.html#region-endpoints
    // for more information.
    static final String HOST = "email-smtp.us-west-2.amazonaws.com";

    // The port you will connect to on the Amazon SES SMTP endpoint.
    static final int PORT = 587;

    static final String SUBJECT = "Amazon SES test (SMTP interface accessed using Java)";

    static final String BODY = String.join(System.getProperty("line.separator"),
        "<h1>Amazon SES SMTP Email Test</h1>",
        "<p>This email was sent with Amazon SES using the ",
        "<a href='https://github.com/javaee/javamail'>Javamail Package</a>",
        " for <a href='https://www.java.com'>Java</a>.");

    public static void main(String[] args) throws Exception {
        // Create a Properties object to contain connection configuration information.
        Properties props = System.getProperties();
        props.put("mail.transport.protocol", "smtp");
        props.put("mail.smtp.port", PORT);
        props.put("mail.smtp.auth", "true");
    }
}
```
// Create a Session object to represent a mail session with the specified properties.
Session session = Session.getDefaultInstance(props);

// Create a message with the specified information.
MimeMessage msg = new MimeMessage(session);
msg.setFrom(new InternetAddress(FROM,FROMNAME));
msg.setRecipient(Message.RecipientType.TO, new InternetAddress(TO));
msg.setSubject(SUBJECT);
msg.setContent(BODY,"text/html");

// Add a configuration set header. Comment or delete the
// next line if you are not using a configuration set
msg.setHeader("X-SES-CONFIGURATION-SET", CONFIGSET);

// Create a transport.
Transport transport = session.getTransport();

// Send the message.
try
{
    System.out.println("Sending...");
    // Connect to Amazon SES using the SMTP username and password you specified
    // above.
    transport.connect(HOST, SMTP_USERNAME, SMTP_PASSWORD);
    // Send the email.
    transport.sendMessage(msg, msg.getAllRecipients());
    System.out.println("Email sent!");
}
catch (Exception ex) {
    System.out.println("The email was not sent.");
    System.out.println("Error message: " + ex.getMessage());
}
finally
{
    // Close and terminate the connection.
    transport.close();
}
}

JavaScript

The following code example is a complete solution for sending email through the Amazon SES SMTP interface by using the NodeMailer module in Node.js.

In order to run this code example, you first have to obtain SMTP credentials. For more information, see Obtaining your Amazon SES SMTP credentials (p. 86). You must also install the NodeMailer module.

/*
This code uses callbacks to handle asynchronous function responses.
It currently demonstrates using an async-await pattern.
AWS supports both the async-await and promises patterns.
For more information, see the following:
https://docs.aws.amazon.com/sdk-for-javascript/v2/developer-guide/calling-services-asynchronously.html
https://docs.aws.amazon.com/lambda/latest/dg/nodejs-prog-model-handler.html
*/
"use strict";
const nodemailer = require("nodemailer");

// If you're using Amazon SES in a region other than US West (Oregon),
// replace email-smtp.us-west-2.amazonaws.com with the Amazon SES SMTP
// endpoint in the appropriate AWS Region.
const smtpEndpoint = "email-smtp.us-west-2.amazonaws.com";

// The port to use when connecting to the SMTP server.
const port = 587;

// Replace sender@example.com with your "From" address.
// This address must be verified with Amazon SES.
const senderAddress = "Mary Major <sender@example.com>";

// Replace recipient@example.com with a "To" address. If your account
// is still in the sandbox, this address must be verified. To specify
// multiple addresses, separate each address with a comma.
var toAddresses = "recipient@example.com";

// CC and BCC addresses. If your account is in the sandbox, these
// addresses have to be verified. To specify multiple addresses, separate
// each address with a comma.
var ccAddresses = "cc-recipient0@example.com,cc-recipient1@example.com";
var bccAddresses = "bcc-recipient@example.com";

// Replace smtp_username with your Amazon SES SMTP user name.
const smtpUsername = "AKIAIOSFODNN7EXAMPLE";

// Replace smtp_password with your Amazon SES SMTP password.
const smtpPassword = "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY";

// (Optional) the name of a configuration set to use for this message.
var configurationSet = "ConfigSet";

// The subject line of the email
var subject = "Amazon SES test (Nodemailer)";

// The email body for recipients with non-HTML email clients.
var body_text = `Amazon SES Test (Nodemailer)
---------------------------------
This email was sent through the Amazon SES SMTP interface using Nodemailer.
`;

// The body of the email for recipients whose email clients support HTML content.
var body_html = `<html>
<head></head>
<body>
<h1>Amazon SES Test (Nodemailer)</h1>
<p>This email was sent with <a href='https://aws.amazon.com/ses/'>Amazon SES</a> using <a href='https://nodemailer.com'>Nodemailer</a> for Node.js.</p>
</body>
</html>`;

// The message tags that you want to apply to the email.
var tag0 = "key0=value0";
var tag1 = "key1=value1";

async function main(){
  // Create the SMTP transport.
  let transporter = nodemailer.createTransport({
    host: smtpEndpoint,
    port: port,
  });

  // Send email using the SMTP transport.
  let info = await transporter.sendMail({
    from: senderAddress,
    to: toAddresses,
    cc: ccAddresses,
    bcc: bccAddresses,
    subject: subject,
    text: body_text,
    html: body_html,
    config: {configurationSet},
    tags: [tag0, tag1],
  });

  console.log(`Message sent: ${info.messageId}`);

  // Add code here to verify that the email was sent.
}

// Close the transport.
transporter.close();
Sending email using the SMTP Interface

```javascript
secure: false, // true for 465, false for other ports
auth: {
  user: smtpUsername,
  pass: smtpPassword
};

// Specify the fields in the email.
let mailOptions = {
  from: senderAddress,
  to: toAddresses,
  subject: subject,
  cc: ccAddresses,
  bcc: bccAddresses,
  text: body_text,
  html: body_html,
  // Custom headers for configuration set and message tags.
  headers: {
    'X-SES-CONFIGURATION-SET': configurationSet,
    'X-SES-MESSAGE-TAGS': tag0,
    'X-SES-MESSAGE-TAGS': tag1
  }
};

// Send the email.
let info = await transporter.sendMail(mailOptions)

console.log("Message sent! Message ID: ", info.messageId);
}
main().catch(console.error);
```

Perl

The following code example is a complete solution for sending email through the Amazon SES SMTP interface using Perl. In order to run this code example, you must obtain SMTP credentials; for more information, see Obtaining your Amazon SES SMTP credentials (p. 86). You must also install the Email::Sender, Email::MIME, and Try::Tiny modules from CPAN.

```perl
#!/usr/bin/perl
use warnings;
use strict;
use Email::Sender::Simple qw(sendmail);
use Email::Sender::Transport::SMTP;
use Email::MIME;
use Try::Tiny;

# Replace sender@example.com with your "From" address.
# This address must be verified.
my $sender = 'Sender name <sender@example.com>';

# Replace recipient@example.com with a "To" address. If your account
# is still in the sandbox, this address must be verified.
my $recipient = 'recipient@example.com';

# Replace smtp_username with your Amazon SES SMTP user name.
my $smtp_username = 'smtp_username';

# Replace smtp_password with your Amazon SES SMTP password.
my $smtp_password = 'smtp_password';

# (Optional) the name of a configuration set to use for this message.
# If you comment out this line, you also need to remove or comment out
```
Sending email using the SMTP Interface

```perl
# the "X-SES-CONFIGURATION-SET:" header below.
my $configset = "ConfigSet";

# If you're using Amazon SES in an AWS Region other than US West (Oregon),
# replace email-smtp.us-west-2.amazonaws.com with the Amazon SES SMTP
# endpoint in the appropriate region.
my $host = "email-smtp.us-west-2.amazonaws.com";
my $port = 587;

# The subject line of the email.
my $subject = "Amazon SES Test (Perl)";

# The HTML body for the email.
my $htmlbody = "<html>
<head></head>
<body>
<h1>Amazon SES SMTP Email Test</h1>
<p>This email was sent with Amazon SES using the
<a href='https://www.perl.org/'>Perl</a>
<a href='http://search.cpan.org/~rjbs/Email-Sender-1.300031/'>Email::Sender</a> library.</p>
</body>
</html>
END_HTML"

# The email body for recipients with non-HTML email clients.
my $textbody = "Amazon SES Test
This message was sent with Amazon SES using the Perl "
"Email::Sender module."

# Create the SMTP transport.
my $transport = Email::Sender::Transport::SMTP->new(
    host => "$host",
    port => "$port",
    ssl => 'starttls',
    sasl_username => "$smtp_username",
    sasl_password => "$smtp_password",
);

# Build a multipart MIME message with an HTML part and a text part.
my $message = Email::MIME->create(
    attributes => { content_type => 'multipart/alternative',
                    charset => 'UTF-8' },
    header_str => { From => "$sender",
                    To => "$recipient",
                    Subject => "$subject" },
    parts => [
        Email::MIME->create(
            attributes => { content_type => 'text/plain' },
            body => "$textbody",
        ),
        Email::MIME->create(
            attributes => { content_type => 'text/html' },
            body => "$htmlbody",
        )
    ],
);

# Add the configuration set header to the MIME message.
$message->header_str_set( 'X-SES-CONFIGURATION-SET' => "$configset" );
```
Sending email using the SMTP Interface

The following code example is a complete solution for sending email through the Amazon SES SMTP interface using PHP. In order to run this code example, you must obtain SMTP credentials; for more information, see Obtaining your Amazon SES SMTP credentials (p. 86). You must also install the PHPMailer package using Composer.

```php
<?php

// Import PHPMailer classes into the global namespace
// These must be at the top of your script, not inside a function
use PHPMailer\PHPMailer\PHPMailer;
use PHPMailer\PHPMailer\Exception;

// If necessary, modify the path in the require statement below to refer to the location of your Composer autoload.php file.
require './vendor/autoload.php';

// Replace sender@example.com with your "From" address.
// This address must be verified with Amazon SES.
$sender = 'sender@example.com';
$senderName = 'Sender Name';

// Replace recipient@example.com with a "To" address. If your account is still in the sandbox, this address must be verified.
$recipient = 'recipient@example.com';

// Replace smtp_username with your Amazon SES SMTP user name.
$usernameSmtp = 'smtp_username';

// Replace smtp_password with your Amazon SES SMTP password.
$passwordSmtp = 'smtp_password';

// Specify a configuration set. If you do not want to use a configuration set, comment or remove the next line.
$configurationSet = 'ConfigSet';

// If you're using Amazon SES in a region other than US West (Oregon), replace email-smtp.us-west-2.amazonaws.com with the Amazon SES SMTP endpoint in the appropriate region.
$host = 'email-smtp.us-west-2.amazonaws.com';
$port = 587;

// The subject line of the email
$subject = 'Amazon SES test (SMTP interface accessed using PHP)';

// The plain-text body of the email
$bodyText = 'Email Test
This email was sent through the Amazon SES SMTP interface using the PHPMailer class.';

// The HTML-formatted body of the email
$bodyHtml = '<h1>Email Test</h1>
<p>This email was sent through the Amazon SES SMTP interface using the PHPMailer class.</p>

// If necessary, comment out the following line in the catch block
// die "Error sending email: $_";

// If something goes wrong, print an error message.
try {
    sendmail($message, { transport => $transport });
}

// Try to send the email using the sendmail function from Email::Sender::Simple.
try {
    sendmail($message, { transport => $transport });
    } catch {
        die "Error sending email: $_";
    }

```
PHPMailer class.

$mail = new PHPMailer(true);

try {
    // Specify the SMTP settings.
    $mail->isSMTP();
    $mail->setFrom($sender, $senderName);
    $mail->Username   = $usernameSmtp;
    $mail->Password   = $passwordSmtp;
    $mail->Host       = $host;
    $mail->Port       = $port;
    $mail->SMTPAuth   = true;
    $mail->SMTPSecure = 'tls';
    $mail->addCustomHeader('X-SES-CONFIGURATION-SET', $configurationSet);

    // Specify the message recipients.
    $mail->addAddress($recipient);
    // You can also add CC, BCC, and additional To recipients here.

    // Specify the content of the message.
    $mail->isHTML(true);
    $mail->Subject    = $subject;
    $mail->Body       = $bodyHtml;
    $mail->AltBody    = $bodyText;
    $mail->Send();
    echo "Email sent! \n", PHP_EOL;
} catch (phpmailerException $e) {
    echo "An error occurred. \n$e->errorMessage()\n", PHP_EOL; //Catch errors from PHPMailer.
} catch (Exception $e) {
    echo "Email not sent. \n$mail->ErrorInfo\n", PHP_EOL; //Catch errors from Amazon SES.
}

?>

Python

The following code example is a complete solution for sending email through the Amazon SES SMTP interface using Python. In order to run this code example, you must obtain SMTP credentials; for more information, see Obtaining your Amazon SES SMTP credentials (p. 86).

```python
import smtplib
import email.utils
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText

# Replace sender@example.com with your "From" address.
# This address must be verified.
SENDER = 'sender@example.com'
SENDERNAME = 'Sender Name'

# Replace recipient@example.com with a "To" address. If your account # is still in the sandbox, this address must be verified.
RECIPIENT = 'recipient@example.com'

# Replace smtp_username with your Amazon SES SMTP user name.
USERNAME_SMTP = "smtp_username"

# Replace smtp_password with your Amazon SES SMTP password.
PASSWORD_SMTP = "smtp_password"

# (Optional) the name of a configuration set to use for this message.
```
# If you comment out this line, you also need to remove or comment out
# the "X-SES-CONFIGURATION-SET:" header below.

CONFIGURATION_SET = "ConfigSet"

# If you're using Amazon SES in an AWS Region other than US West (Oregon),
# replace email-smtp.us-west-2.amazonaws.com with the Amazon SES SMTP
# endpoint in the appropriate region.

HOST = "email-smtp.us-west-2.amazonaws.com"
PORT = 587

# The subject line of the email.
SUBJECT = 'Amazon SES Test (Python smtplib)'

# The email body for recipients with non-HTML email clients.

BODY_TEXT = ("Amazon SES Test\r\n\n" +
              "This email was sent through the Amazon SES SMTP "
              + "Interface using the Python smtplib package."
            )

# The HTML body of the email.
BODY_HTML = """"""""""""""""""""""""""""

# Create message container - the correct MIME type is multipart/alternative.
msg = MIMEMultipart('alternative')
msg['Subject'] = SUBJECT
msg['From'] = email.utils.formataddr((SENDERNAME, SENDER))
msg['To'] = RECIPIENT
# Comment or delete the next line if you are not using a configuration set
msg.add_header('X-SES-CONFIGURATION-SET',CONFIGURATION_SET)
# Record the MIME types of both parts - text/plain and text/html.
part1 = MIMEText(BODY_TEXT, 'plain')
part2 = MIMEText(BODY_HTML, 'html')
# Attach parts into message container.
# According to RFC 2046, the last part of a multipart message, in this case
# the HTML message, is best and preferred.
msg.attach(part1)
msg.attach(part2)

# Try to send the message.
try:
    server = smtplib.SMTP(HOST, PORT)
    server.ehlo()
    server.starttls()
    server.login(USERNAME_SMTP, PASSWORD_SMTP)
    server.sendmail(SENDER, RECIPIENT, msg.as_string())
    server.close()
except Exception as e:
    print ("Error: ", e)
else:
    print ("Email sent!")
Ruby

The following code example is a complete solution for sending email through the Amazon SES SMTP interface using Ruby. In order to run this code example, you must obtain SMTP credentials; for more information, see Obtaining your Amazon SES SMTP credentials (p. 86).

```ruby
require 'net/smtp'

# Replace sender@example.com with your "From" address.
# This address must be verified with Amazon SES.
sender = "sender@example.com"
senderName = "Sender Name"

# Replace recipient@example.com with a "To" address. If your account
# is still in the sandbox, this address must be verified.
recipient = "recipient@example.com"

# Replace smtp_username with your Amazon SES SMTP user name.
smtp_username = "smtp_username"

# Replace smtp_password with your Amazon SES SMTP password.
smtp_password = "smtp_password"

# (Optional) the name of a configuration set to use for this message.
# If you comment out this line, you also need to remove or comment out
# the "X-SES-CONFIGURATION-SET" header below.
configSet = "ConfigSet"

# If you're using Amazon SES in an AWS Region other than US West (Oregon),
# replace email-smtp.us-west-2.amazonaws.com with the Amazon SES SMTP
# endpoint in the appropriate region.
server = "email-smtp.us-west-2.amazonaws.com"
port = 587

# The subject line of the email.
subject = "Amazon SES Test (Ruby Net::SMTP library)"

# Specify the headers and body of the message as a variable.
message = [
  #Remove the next line if you are not using a configuration set
  "X-SES-CONFIGURATION-SET: #{configSet}",
  "Content-Type: text/html; charset=UTF-8",
  "Content-Transfer-Encoding: 7bit",
  "From: #{senderName} <#{sender}>",
  "To: #{recipient}",
  "Subject: #{subject}",
  "",
  "<h1>Amazon SES Test (Ruby Net::SMTP library)</h1>",
  "<p>This email was sent with \n  <a href='https://aws.amazon.com/ses/'>Amazon SES</a> using the Ruby Net::SMTP library.</p>"]

# Create a new SMTP object called "smtp."
smtp = Net::SMTP.new(server, port)

# Tell the smtp object to connect using TLS.
smtp.enable_starttls

# Open an SMTP session and log in to the server using SMTP authentication.
smtp.start(server, smtp_username, smtp_password, :login)

# Try to send the message.
beginsmtp.send_message(message, sender, recipient)
```

441
Sending raw email using AWS SDKs

The AWS SDKs contain built-in methods for interacting with Amazon SES and several other AWS services. If you plan to use Amazon SES along with other AWS services, we recommend that you use an SDK. To learn more about the AWS SDKs, see Tools for Amazon Web Services.

In this section, you will find code examples in several programming languages that demonstrate the process of sending raw email through Amazon SES using the AWS SDKs.

Java

The following code example shows how to use the JavaMail library and the AWS SDK for Java to compose and send a raw email that contains an HTML part, a text part, and an attachment.

This code example assumes you have installed the AWS SDK for Java, and that you have created a shared credentials file. For more information about creating a shared credentials file, see Create a shared credentials file (p. 31).

Important
You use a shared credentials file to pass your AWS access key ID and secret access key. As an alternative to using a shared credentials file, you can specify your AWS access key ID and secret access key by setting two environment variables (AWS_ACCESS_KEY_ID and AWS_SECRET_ACCESS_KEY, respectively). This example doesn't function unless you specify your credentials using one of these methods.

```
package com.amazonaws.samples;

import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.io.PrintStream;
import java.nio.ByteBuffer;
import java.util.Properties;
// JavaMail libraries. Download the JavaMail API
// from https://javaee.github.io/javamail/
import javax.activation.DataHandler;
import javax.activation.DataSource;
import javax.activation.FileDataSource;
import javax.mail.Message;
import javax.mail.MessagingException;
import javax.mail.Session;
import javax.mail.internet.AddressException;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeBodyPart;
import javax.mail.internet.MimeMultipart;
// AWS SDK libraries. Download the AWS SDK for Java
// from https://aws.amazon.com/sdk-for-java
import com.amazonaws.regions.Regions;
import com.amazonaws.services.simpleemail.AmazonSimpleEmailService;
import com.amazonaws.services.simpleemail.AmazonSimpleEmailServiceClientBuilder;
import com.amazonaws.services.simpleemail.model.RawMessage;
import com.amazonaws.services.simpleemail.model.SendRawEmailRequest;
```
public class AmazonSESSample {

    // Replace sender@example.com with your "From" address.
    // This address must be verified with Amazon SES.
    private static String SENDER = "Sender Name <sender@example.com>";

    // Replace recipient@example.com with a "To" address. If your account
    // is still in the sandbox, this address must be verified.
    private static String RECIPIENT = "recipient@example.com";

    // Specify a configuration set. If you do not want to use a configuration
    // set, comment the following variable, and the
    // ConfigurationSetName=CONFIGURATION_SET argument below.
    private static String CONFIGURATION_SET = "ConfigSet";

    // The subject line for the email.
    private static String SUBJECT = "Customer service contact info";

    // The full path to the file that will be attached to the email.
    // If you're using Windows, escape backslashes as shown in this variable.
    private static String ATTACHMENT = "C:\Users\sender\customers-to-contact.xlsx";

    // The email body for recipients with non-HTML email clients.
    private static String BODY_TEXT = "Hello,
    + "Please see the attached file for a list "+ "of customers to contact."

    // The HTML body of the email.
    private static String BODY_HTML = "<html>
    + "<head></head>"
    + "<body>"
    + "<h1>Hello!</h1>"
    + "<p>Please see the attached file for a "
    + "list of customers to contact."</p>"
    + "</body>"
    + "</html>";

    public static void main(String[] args) throws AddressException, MessagingException, IOException {

        Session session = Session.getDefaultInstance(new Properties());

        // Create a new MimeMessage object.
        MimeMessage message = new MimeMessage(session);

        // Add subject, from and to lines.
        message.setSubject(SUBJECT, "UTF-8");
        message.setFrom(new InternetAddress(SENDER));
        message.setRecipients(Message.RecipientType.TO, InternetAddress.parse(RECIPIENT));

        // Create a multipart/alternative child container.
        MimeMultipart msg_body = new MimeMultipart("alternative");

        // Create a wrapper for the HTML and text parts.
        MimeBodyPart wrap = new MimeBodyPart();

        // Define the text part.
        MimeBodyPart textPart = new MimeBodyPart();
        textPart.setContent(BODY_TEXT, "text/plain; charset=UTF-8");

        // Define the HTML part.
        MimeBodyPart htmlPart = new MimeBodyPart();
        htmlPart.setContent(BODY_HTML, "text/html; charset=UTF-8");

        // Add the text and HTML parts to the child container.
msg_body.addBodyPart(textPart);
msg_body.addBodyPart(htmlPart);

// Add the child container to the wrapper object.
wrap.setContent(msg_body);

// Create a multipart/mixed parent container.
MimeMultipart msg = new MimeMultipart("mixed");

// Add the parent container to the message.
message.setContent(msg);

// Add the multipart/alternative part to the message.
msg.addBodyPart(wrap);

// Define the attachment
MimeBodyPart att = new MimeBodyPart();
DataSource fds = new FileDataSource(ATTACHMENT);
att.setDataHandler(new DataHandler(fds));
att.setFileName(fds.getName());

// Add the attachment to the message.
msg.addBodyPart(att);

// Try to send the email.
try {
    System.out.println("Attempting to send an email through Amazon SES " + "using the AWS SDK for Java...”);

    // Instantiate an Amazon SES client, which will make the service
    // call with the supplied AWS credentials.
    AmazonSimpleEmailService client =
        AmazonSimpleEmailServiceClientBuilder.standard()
        .withRegion(Regions.US_WEST_2).build();

    // Print the raw email content on the console
    PrintStream out = System.out;
    message.writeTo(out);

    // Send the email.
    ByteArrayOutputStream outputStream = new ByteArrayOutputStream();
    message.writeTo(outputStream);
    RawMessage rawMessage =
        new RawMessage(ByteBuffer.wrap(outputStream.toByteArray()));
    SendRawEmailRequest rawEmailRequest =
        new SendRawEmailRequest(rawMessage)
        .withConfigurationSetName(CONFIGURATION_SET);

    client.sendRawEmail(rawEmailRequest);
    System.out.println("Email sent!");
    // Display an error if something goes wrong.
} catch (Exception ex) {
    System.out.println("Email Failed");
    System.err.println("Error message: " + ex.getMessage());
    ex.printStackTrace();
}
}
The following code example shows how to use the \texttt{PHPMailer} package and the \texttt{AWS SDK for PHP} to compose and send a raw email that contains an HTML part, a text part, and an attachment.

This code example assumes that you have installed the \texttt{PHPMailer} package using \texttt{Composer}. It also assumes that you have installed the AWS SDK for PHP, and that you have created a shared credentials file. For more information about creating a shared credentials file, see \texttt{Create a shared credentials file} (p. 31).

\textbf{Important}

You use a shared credentials file to pass your AWS access key ID and secret access key. As an alternative to using a shared credentials file, you can specify your AWS access key ID and secret access key by setting two environment variables (\texttt{AWS_ACCESS_KEY_ID} and \texttt{AWS_SECRET_ACCESS_KEY}, respectively). This example doesn't function unless you specify your credentials using one of these methods.

```php
<?php
require 'vendor/autoload.php';
use PHPMailer\PHPMailer\PHPMailer;
use Aws\Ses\SesClient;
use Aws\Ses\Exception\SesException;

// Replace sender@example.com with your "From" address.
// This address must be verified with Amazon SES.
$sender = 'sender@example.com';
$sendername = 'Sender Name';

// Replace recipient@example.com with a "To" address. If your account
// is still in the sandbox, this address must be verified.
$recipient = 'recipient@example.com';

// Specify a configuration set.
#configset = 'ConfigSet';

// Replace us-west-2 with the AWS Region you're using for Amazon SES.
#region = 'us-west-2';

#subject = 'List of customers to contact';

#htmlbody = <<<EOD
<html>
<head></head>
<body>
<h1>Hello!</h1>
<p>Please see the attached file for a list of customers to contact.</p>
</body>
</html>
EOD;

#textbody = <<<EOD
Hello,
Please see the attached file for a list of customers to contact.
EOD;

// The full path to the file that will be attached to the email.
#att = 'path/to/customers-to-contact.xlsx';

// Create a SesClient.
$client = SesClient::factory(array(
    'version' => 'latest',
    'region' => $region
));
```
// Create a new PHPMailer object.
$mail = new PHPMailer;

// Add components to the email.
$mail->setFrom($sender, $sendername);
$mail->addAddress($recipient);
$mail->Subject = $subject;
$mail->Body = $htmlbody;
$mail->AltBody = $textbody;
$mail->addAttachment($att);
$mail->addCustomHeader('X-SES-CONFIGURATION-SET', $configset);

// Attempt to assemble the above components into a MIME message.
if (!$mail->preSend()) {
    echo $mail->ErrorInfo;
} else {
    // Create a new variable that contains the MIME message.
    $message = $mail->getSentMIMEMessage();
}

// Try to send the message.
try {
    $result = $client->sendRawEmail(
        ['RawMessage' => 
            ['Data' => $message]
        ]);;
    // If the message was sent, show the message ID.
    $messageId = $result->get('MessageId');
    echo("Email sent! Message ID: 

    // If the message was not sent, show a message explaining what went wrong.
    echo("The email was not sent. Error message: 

Python

The following code example shows how to use the Python email package and the AWS SDK for Python (Boto) to compose and send a raw email that contains an HTML part, a text part, and an attachment.

This code example assumes that you have installed the AWS SDK for Python (Boto), and that you have created a shared credentials file. For more information about creating a shared credentials file, see Create a shared credentials file (p. 31).

**Important**

You use a shared credentials file to pass your AWS access key ID and secret access key. As an alternative to using a shared credentials file, you can specify your AWS access key ID and secret access key by setting two environment variables [AWS_ACCESS_KEY_ID and AWS_SECRET_ACCESS_KEY, respectively]. This example doesn't function unless you specify your credentials using one of these methods.

```python
import os
import boto3
from botocore.exceptions import ClientError
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.application import MIMEApplication

# Replace sender@example.com with your "From" address.
```
# This address must be verified with Amazon SES.
SENDER = "Sender Name <sender@example.com>"

# Replace recipient@example.com with a "To" address. If your account
# is still in the sandbox, this address must be verified.
RECIPIENT = "recipient@example.com"

# Specify a configuration set. If you do not want to use a configuration
# set, comment the following variable, and the
# ConfigurationSetName=CONFIGURATION_SET argument below.
CONFIGURATION_SET = "ConfigSet"

# If necessary, replace us-west-2 with the AWS Region you’re using for Amazon SES.
AWS_REGION = "us-west-2"

# The subject line for the email.
SUBJECT = "Customer service contact info"

# The full path to the file that will be attached to the email.
ATTACHMENT = "path/to/customers-to-contact.xlsx"

# The email body for recipients with non-HTML email clients.
BODY_TEXT = "Hello,
Please see the attached file for a list of customers to contact."

# The HTML body of the email.
BODY_HTML = """"<html>
<head></head>
<body>
<h1>Hello!</h1>
<p>Please see the attached file for a list of customers to contact.</p>
</body>
</html>"

# The character encoding for the email.
CHARSET = "utf-8"

# Create a new SES resource and specify a region.
client = boto3.client('ses', region_name=AWS_REGION)

# Create a multipart/mixed parent container.
msg = MIME_Multipart('mixed')
# Add subject, from and to lines.
msg['Subject'] = SUBJECT
msg['From'] = SENDER
msg['To'] = RECIPIENT

# Create a multipart/alternative child container.
msg_body = MIME_Multipart('alternative')

# Encode the text and HTML content and set the character encoding. This step is
# necessary if you’re sending a message with characters outside the ASCII range.
textpart = MIME_Text(BODY_TEXT.encode(CHARSET), 'plain', CHARSET)
htmlpart = MIME_Text(BODY_HTML.encode(CHARSET), 'html', CHARSET)

# Add the text and HTML parts to the child container.
msg_body.attach(textpart)
msg_body.attach(htmlpart)

# Define the attachment part and encode it using MIMEApplication.
att = MIME_Application(open(ATTACHMENT, 'rb').read())

# Add a header to tell the email client to treat this part as an attachment,
# and to give the attachment a name.
att.add_header('Content-Disposition', 'attachment', filename=os.path.basename(ATTACHMENT))

# Attach the multipart/alternative child container to the multipart/mixed parent container.
msg.attach(msg_body)

# Add the attachment to the parent container.
msg.attach(att)

try:
    # Provide the contents of the email.
    response = client.send_raw_email(
        Source=SENDER,
        Destinations=[
            RECIPIENT
        ],
        RawMessage={
            'Data':msg.as_string(),
        },
        ConfigurationSetName=CONFIGURATION_SET
    )
    # Display an error if something goes wrong.
    except ClientError as e:
        print(e.response['Error']['Message'])
    else:
        print("Email sent! Message ID:",
              print(response['MessageId'])

Ruby

The following code example shows how to use the Ruby MIME gem and the AWS SDK for Ruby to compose and send a raw email that contains an HTML part, a text part, and an attachment.

This code example assumes that you have installed the AWS SDK for Ruby and the MIME gem, and that you have created a shared credentials file. For more information about creating a shared credentials file, see Create a shared credentials file (p. 31).

Important

You use a shared credentials file to pass your AWS access key ID and secret access key. As an alternative to using a shared credentials file, you can specify your AWS access key ID and secret access key by setting two environment variables (AWS_ACCESS_KEY_ID and AWS_SECRET_ACCESS_KEY, respectively). This example doesn't function unless you specify your credentials using one of these methods.

require 'base64'  # standard library
require 'aws-sdk' # gem install aws-sdk
require 'mime'    # gem install mime

# Replace sender@example.com with your "From" address.
# This address must be verified with Amazon SES.
sender = "sender@example.com"
sendername = "Sender Name"

# Replace recipient@example.com with a "To" address. If your account
# is still in the sandbox, this address must be verified.
recipient = "recipient@example.com"

# Specify a configuration set.
configsetname = "ConfigSet"

# Replace us-west-2 with the AWS Region you're using for Amazon SES.
awsregion = "us-west-2"
# The subject line for the email.
subject = "Customer service contact info"

# The full path to the file that will be attached to the email.
attachment = "path/to/customers-to-contact.xlsx"

# The email body for recipients with non-HTML email clients.
textbody = """Hello,
Please see the attached file for a list of customers to contact.
"

# The HTML body of the email.
htmlbody = """<html>
<head></head>
<body>
<h1>Hello!</h1>
<p>Please see the attached file for a list of customers to contact.</p>
</body>
</html>
""

# Create a new MIME text object that contains the base64-encoded content of the
# file that will be attached to the message.
file = MIME::Application.new(Base64::encode64(open(attachment,"rb").read))

# Specify that the file is a base64-encoded attachment to ensure that the
# receiving client handles it correctly.
file.transfer_encoding = 'base64'
file.disposition = 'attachment'

# Create a MIME Multipart Mixed object. This object will contain the body of the
# email and the attachment.
msg_mixed = MIME::Multipart::Mixed.new

# Create a MIME Multipart Alternative object. This object will contain both the
# HTML and plain text versions of the email.
msg_body = MIME::Multipart::Alternative.new

# Add the plain text and HTML content to the Multipart Alternative part.
msg_body.add(MIME::Text.new(textbody,'plain'))
msg_body.add(MIME::Text.new(htmlbody,'html'))

# Add the Multipart Alternative part to the Multipart Mixed part.
msg_mixed.add(msg_body)

# Add the attachment to the Multipart Mixed part.
msg_mixed.attach(file, 'filename' => attachment)

# Create a new Mail object that contains the entire Multipart Mixed object.
# This object also contains the message headers.
msg = MIME::Mail.new(msg_mixed)
msg.to = { recipient => nil }
msg.from = { sender => sendername }
msg.subject = subject
msg.headers.set('X-SES-CONFIGURATION-SET',configsetname)

# Create a new SES resource and specify a region
ses = Aws::SES::Client.new(region: awsregion)

# Try to send the email.
begin

# Provide the contents of the email.
resp = ses.send_raw_email({}
Verify multiple email addresses

If you are migrating to Amazon SES from another email-sending solution, you may already have a long list of email addresses that you want to use to send email. The Python script in this example accepts a JSON-formatted list of email addresses as an input. The following example shows the structure of the input file:

```json
[
  {
    "email":"carlos.salazar@example.com"
  },
  {
    "email":"mary.major@example.co.uk"
  },
  {
    "email":"wei.zhang@example.cn"
  }
]
```

The following script reads the input file and attempts to validate all of the email addresses contained in the file. This code example assumes that you have installed the AWS SDK for Python (Boto), and that you have created a shared credentials file. For more information about creating a shared credentials file, see Create a shared credentials file (p. 31).

```python
import json     # Python standard library
import boto3    # sudo pip install boto3
from botocore.exceptions import ClientError

# The full path to the file that contains the identities to be verified.
# The input file must be JSON-formatted. See
# for a sample input file.
FILE_INPUT = '/path/to/identities.json'

# If necessary, replace us-west-2 with the AWS Region you're using for Amazon SES.
AWS_REGION = "us-west-2"

# Create a new SES resource specify a region.
client = boto3.client('ses', region_name=AWS_REGION)

# Read the file that contains the identities to be verified.
with open(FILE_INPUT) as data_file:
    data = json.load(data_file)

# Iterate through the array from the input file. Each time an object named
# 'email' is found, run the verify_email_identity operation against the value
# of that object.
```
Replicating email identities across AWS Regions

If you want to use the same identity to send email in more than one AWS Region, you have to verify that identity in each Region. The Python script in this example simplifies the process of verifying an identity in a new Region if you've already verified it in a different Region.

You can use this script to replicate either email address or domain identities. If you use this script to replicate an email address, you have to click the verification link in the email that Amazon SES sends to that address. You can only use this script to replicate a domain identity if the DNS settings for that domain are managed by Route 53.

Note
Your sandbox and account quota settings are unique to each AWS Region. In other words, if your account has been removed from the Amazon SES sandbox in one Region, you still have to have it removed in each other Region. For more information about having your account removed from the Amazon SES sandbox, see Moving out of the Amazon SES sandbox (p. 71). For more information about increasing your Amazon SES service quotas, see Increasing your Amazon SES sending quotas (p. 146).

```python
import argparse
import json
import logging
from pprint import pprint
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

def get_identities(ses_client):
    
    email_identities = []
    domain_identities = []

    try:
        identity_paginator = ses_client.get_paginator('list_identities')
        identity_iterator = identity_paginator.paginate(
            PaginationConfig={'PageSize': 20})
        for identity_page in identity_iterator:
            for identity in identity_page['Identities']:
                if '@' in identity:
                    email_identities.append(identity)
                else:
                    domain_identities.append(identity)

        return email_identities, domain_identities
    except ClientError as e:
        print(e.response['Error']['Message'])
```

```python
for i in data:
    try:
        response = client.verify_email_identity(
            EmailAddress=i['email'])
    )
    
    # Display an error if something goes wrong.
    except ClientError as e:
        print(e.response['Error']['Message'])
    
    # Otherwise, show the request ID of the verification message.
    else:
        print('Verification email sent to ' + i['email'] + '. Request ID: ' +
            response['ResponseMetadata']['RequestId'])
```
else:
    domain_identities.append(identity)
    logger.info("Found %s email and %s domain identities.", len(email_identities), len(domain_identities))
except ClientError:
    logger.exception("Couldn't get identities.")
    raise
else:
    return email_identities, domain_identities

def verify_emails(email_list, ses_client):
    ""
    Starts verification of a list of email addresses. Verification causes an email
to be sent to each address. To complete verification, the recipient must follow
the instructions in the email.

    :param email_list: The list of email addresses to verify.
    :return: The list of emails that were successfully submitted for verification.
    ""
    verified_emails = []
    for email in email_list:
        try:
            ses_client.verify_email_identity(EmailAddress=email)
            verified_emails.append(email)
            logger.info("Started verification of %s.", email)
        except ClientError:
            logger.warning("Couldn't start verification of %s.", email)
    return verified_emails

def verify_domains(domain_list, ses_client):
    ""
    Starts verification for a list of domain identities. This returns a token for
each domain, which must be registered as a TXT record with the DNS provider for
the domain.

    :param domain_list: The list of domains to verify.
    :return: The generated domain tokens to use to completed verification.
    ""
    domain_tokens = {}
    for domain in domain_list:
        try:
            response = ses_client.verify_domain_identity(Domain=domain)
            token = response['VerificationToken']
            domain_tokens[domain] = token
            logger.info("Got verification token %s for domain %s.", token, domain)
        except ClientError:
            logger.warning("Couldn't get verification token for domain %s.", domain)
    return domain_tokens

def get_hosted_zones(route53_client):
    ""
    Gets the Amazon Route 53 hosted zones for the current account.

    :return: The list of hosted zones.
    ""
    zones = []
    try:
        zone Paginator = route53_client.get_paginator('list_hosted_zones')
        zone_iterator = zone Paginator.paginate(PaginationConfig={'PageSize': 20})
        for page in zone_iterate...
Amazon Simple Email Service Classic Developer Guide
Replicating identities
zones = [
zone for zone_page in zone_iterator for zone in zone_page['HostedZones']]
logger.info("Found %s hosted zones.", len(zones))
except ClientError:
logger.warning("Couldn't get hosted zones.")
return zones
def find_domain_zone_matches(domains, zones):
"""
Finds matches between Amazon SES verified domains and Route 53 hosted zones.
Subdomain matches are taken when found, otherwise root domain matches are taken.
:param domains: The list of domains to match.
:param zones: The list of hosted zones to match.
:return: The set of matched domain-zone pairs. When a match is not found, the
domain is included in the set with a zone value of None.
"""
domain_zones = {}
for domain in domains:
domain_zones[domain] = None
# Start at the most specific sub-domain and walk up to the root domain until a
# zone match is found.
domain_split = domain.split('.')
for index in range(0, len(domain_split) - 1):
sub_domain = '.'.join(domain_split[index:])
for zone in zones:
# Normalize the zone name from Route 53 by removing the trailing '.'.
zone_name = zone['Name'][:-1]
if sub_domain == zone_name:
domain_zones[domain] = zone
break
if domain_zones[domain] is not None:
break
return domain_zones
def add_route53_verification_record(domain, token, zone, route53_client):
"""
Adds a domain verification TXT record to the specified Route 53 hosted zone.
When a TXT record already exists in the hosted zone for the specified domain,
the existing values are preserved and the new token is added to the list.
:param domain: The domain to add.
:param token: The verification token for the domain.
:param zone: The hosted zone where the domain verification record is added.
"""
domain_token_record_set_name = f'_amazonses.{domain}'
record_set_paginator = route53_client.get_paginator(
'list_resource_record_sets')
record_set_iterator = record_set_paginator.paginate(
HostedZoneId=zone['Id'], PaginationConfig={'PageSize': 20})
records = []
for record_set_page in record_set_iterator:
try:
txt_record_set = next(
record_set for record_set
in record_set_page['ResourceRecordSets']
if record_set['Name'][:-1] == domain_token_record_set_name and
record_set['Type'] == 'TXT')
records = txt_record_set['ResourceRecords']
logger.info(
"Existing TXT record found in set %s for zone %s.",
domain_token_record_set_name, zone['Name'])
break

453


except StopIteration:
    pass
records.append({'Value': json.dumps(token)})
changes = [{
    'Action': 'UPSERT',
    'ResourceRecordSet': {
        'Name': domain_token_record_set_name,
        'Type': 'TXT',
        'TTL': 1800,
        'ResourceRecords': records}}]
try:
    route53_client.change_resource_record_sets(
        HostedZoneId=zone['Id'], ChangeBatch={'Changes': changes})
    logger.info("Created or updated the TXT record in set %s for zone %s.",
        domain_token_record_set_name, zone['Name'])
except ClientError as err:
    logger.warning("Got error %s. Couldn't create or update the TXT record for zone %s.",
        err.response['Error']['Code'], zone['Name'])

def generate_dkim_tokens(domain, ses_client):
    """
    Generates DKIM tokens for a domain. These must be added as CNAME records to the
    DNS provider for the domain.
    :param domain: The domain to generate tokens for.
    :return: The list of generated DKIM tokens.
    """
    dkim_tokens = []
    try:
        dkim_tokens = ses_client.verify_domain_dkim(Domain=domain)['DkimTokens']
        logger.info("Generated %s DKIM tokens for domain %s.", len(dkim_tokens), domain)
    except ClientError:
        logger.warning("Couldn't generate DKIM tokens for domain %s.", domain)
    return dkim_tokens

def add_dkim_domain_tokens(hosted_zone, domain, tokens, route53_client):
    """
    Adds DKIM domain token CNAME records to a Route 53 hosted zone.
    :param hosted_zone: The hosted zone where the records are added.
    :param domain: The domain to add.
    :param tokens: The DKIM tokens for the domain to add.
    """
    try:
        changes = [{
            'Action': 'UPSERT',
            'ResourceRecordSet': {
                'Name': f'{token}._domainkey.{domain}',
                'Type': 'CNAME',
                'TTL': 1800,
                'ResourceRecords': [{'Value': f'{token}.dkim.amazonses.com'}]
            } for token in tokens]
        route53_client.change_resource_record_sets(
            HostedZoneId=hosted_zone['Id'], ChangeBatch={'Changes': changes})
        logger.info("Added %s DKIM CNAME records to %s in zone %s.", len(tokens),
            domain, hosted_zone['Name'])
    except ClientError:
        logger.warning("Couldn't add DKIM CNAME records for %s to zone %s.", domain,
def configure_sns_topics(identity, topics, ses_client):
    """
    Configures Amazon Simple Notification Service (Amazon SNS) notifications for
    an identity. The Amazon SNS topics must already exist.
    
    :param identity: The identity to configure.
    :param topics: The list of topics to configure. The choices are Bounce, Delivery, or Complaint.
    """
    for topic in topics:
        topic_arn = input(f"Enter the Amazon Resource Name (ARN) of the {topic} topic or press "
                          f"Enter to skip: ")
        if topic_arn != '':
            try:
                ses_client.set_identity_notification_topic(
                    Identity=identity, NotificationType=topic, SnsTopic=topic_arn)
                logger.info("Configured %s for %s notifications.", identity, topic)
            except ClientError:
                logger.warning("Couldn't configure %s for %s notifications.", identity, topic)

def replicate(source_client, destination_client, route53_client):
    logging.basicConfig(level=logging.INFO, format='%(levelname)s: %(message)s')
    print('-'*88)
    print(f"Replicating Amazon SES identities and other configuration from
          {source_client.meta.region_name} to {destination_client.meta.region_name}.")
    print('-'*88)
    print(f"Retrieving identities from {source_client.meta.region_name}.")
    source_emails, source_domains = get_identities(source_client)
    print("Email addresses found:")
    print(*source_emails)
    print("Domains found:")
    print(*source_domains)

    print("Starting verification for email identities.")
    dest_emails = verify_emails(source_emails, destination_client)
    print("Getting domain tokens for domain identities.")
    dest_domain_tokens = verify_domains(source_domains, destination_client)

    # Get Route 53 hosted zones and match them with Amazon SES domains.
    answer = input("Is the DNS configuration for your domains managed by Amazon Route 53 (y/n)? ")
    use_route53 = answer.lower() == 'y'
    hosted_zones = get_hosted_zones(route53_client) if use_route53 else []
    if use_route53:
        print("Adding or updating Route 53 TXT records for your domains.")
        domain_zones = find_domain_zone_matches(dest_domain_tokens.keys(), hosted_zones)
        for domain in domain_zones:
            add_route53_verification_record( 
                domain, dest_domain_tokens[domain], domain_zones[domain], route53_client)
        else:
            print("Use these verification tokens to create TXT records through your DNS 
                "provider:"
                pprint(dest_domain_tokens)

    answer = input("Do you want to configure DKIM signing for your identities (y/n)? ")
    if answer.lower() == 'y':
# Build a set of unique domains from email and domain identities.
domains = {email.split('@')[1] for email in dest_emails}
domains.update(dest_domain_tokens)
domain_zones = find_domain_zone_matches(domains, hosted_zones)
for domain, zone in domain_zones.items():
    answer = input(
        f"Do you want to configure DKIM signing for {domain} (y/n)? ")
    if answer.lower() == 'y':
        dkim_tokens = generate_dkim_tokens(domain, destination_client)
        if use_route53 and zone is not None:
            add_dkim_domain_tokens(zone, domain, dkim_tokens, route53_client)
        else:
            print(
                "Add the following DKIM tokens as CNAME records through your "
                "DNS provider:")
            print(*dkim_tokens, sep='\n')
    answer = input(
        "Do you want to configure Amazon SNS notifications for your identities (y/n)? ")
    if answer.lower() == 'y':
        for identity in dest_emails + list(dest_domain_tokens.keys()):
            answer = input(
                f"Do you want to configure Amazon SNS topics for {identity} (y/n)? ")
            if answer.lower() == 'y':
                configure_sns_topics(
                    identity, ['Bounce', 'Delivery', 'Complaint'], destination_client)
    print(f"Replication complete for {destination_client.meta.region_name}.")
    print('-'*88)

def main():
    boto3_session = boto3.Session()
    ses_regions = boto3_session.get_available_regions('ses')
    parser = argparse.ArgumentParser(
        description="Copies email address and domain identities from one AWS Region to 
        another. Optionally adds records for domain verification and DKIM 
        "signing to domains that are managed by Amazon Route 53, 
        "and sets up Amazon SNS notifications for events of interest.")
    parser.add_argument("source_region", choices=ses_regions, help="The region to copy from.")
    parser.add_argument("destination_region", choices=ses_regions, help="The region to copy to.")
    args = parser.parse_args()
    source_client = boto3.client('ses', region_name=args.source_region)
    destination_client = boto3.client('ses', region_name=args.destination_region)
    route53_client = boto3.client('route53')
    replicate(source_client, destination_client, route53_client)

if __name__ == '__main__':
    main()
Regions and Amazon SES

Amazon SES is available in several AWS Regions around the world. In each Region, AWS maintains multiple Availability Zones. These Availability Zones are physically isolated from each other, but are united by private, low-latency, high-throughput, and highly redundant network connections. These Availability Zones enable us to provide very high levels of availability and redundancy, while also minimizing latency.

For a list of all of the Amazon SES Regional endpoints, see Amazon Simple Email Service endpoints and quotas in the Amazon Web Services General Reference. To learn more about the number of Availability Zones that are available in each Region, see AWS Global Infrastructure.

This section contains information that you need to know if you plan to use Amazon SES in multiple AWS Regions. It discusses the following subjects:

- Amazon SES regions and endpoints (p. 457)
- Sandbox and sending limit increases (p. 458)
- Verification of email addresses and domains (p. 458)
- Easy DKIM (p. 458)
- Suppression list (p. 458)
- Feedback notifications (p. 458)
- SMTP credentials (p. 459)
- Sending authorization (p. 460)
- Custom MAIL FROM domains (p. 459)
- Email receiving (p. 460)

For general information about AWS Regions, see AWS Regions and Endpoints in the AWS General Reference.

Amazon SES regions and endpoints

When you use Amazon Simple Email Service (Amazon SES) to send email, you connect to a URL that provides an endpoint for the Amazon SES API or SMTP interface. The AWS General Reference contains a complete list of endpoints that you use to send and receive email through Amazon SES. For more information, see Amazon Simple Email Service (Amazon SES) in the AWS General Reference.

When you send email through Amazon SES, you can use the URLs in the API (HTTPS) Endpoint column to make HTTPS requests to the Amazon SES API. You can also use the URLs in the SMTP Endpoint column to send email by using the SMTP interface.

If you've configured Amazon SES to receive email that's sent to your domain, you can use the inbound SMTP endpoint URLs (that is, the URLs that begin with "inbound-smtp.") when you set up the mail exchanger (MX) records in the DNS settings for your domain (p. 213).

Note
The inbound SMTP URLs aren't IMAP server addresses. In other words, you can't use them to receive email by using an application such as Outlook. For a service that provides an IMAP server for incoming email, see Amazon WorkMail.
Sandbox and sending limit increases

The sandbox status for your account can differ between AWS Regions. In other words, if your account has been removed from the sandbox in the US West (Oregon) Region, it might still be in the sandbox in the US East (N. Virginia) Region, unless you've also had it removed from the sandbox in that Region.

Sending limits can also be different depending on the AWS Region. For example, if your account is able to send 10 messages per second in the Europe (Ireland) Region, you might be able to send more or fewer messages in other Regions.

When you submit a request to have your account removed from the sandbox (p. 71), or when you submit a request to have your account’s sending quotas increased (p. 147), be sure to choose all of the AWS Regions that your request applies to. You can submit several requests in a single Support Center case.

Verification of email addresses and domains

Before you can send email using Amazon SES, you have to verify that you own the email address or domain that you plan to send from. The verification status of email addresses and domains also differs across AWS Regions. For example, if you verify a domain in the US West (Oregon) Region, you can’t use that domain to send email in the US East (N. Virginia) Region until you complete the verification process again for that Region. For more information about verifying email addresses and domains, see Verifying identities in Amazon SES (p. 47).

Easy DKIM

You have to perform the Easy DKIM setup process for each Region where you want to use Easy DKIM. That is, in each Region, you have to use the Amazon SES console or the Amazon SES API to generate TXT records. Next, you have to add all of the TXT records to the DNS configuration for your domain. For more information about setting up Easy DKIM, see Easy DKIM in Amazon SES (p. 131).

Suppression list

Although each Region has a separate global suppression list, if you remove an address from the global suppression list in one Region, Amazon SES removes the address from the global suppression list in all Regions. You remove addresses from the suppression list by using the Amazon SES console. For more information about removing addresses from the suppression list, see Using the Amazon SES global suppression list (p. 192).

Feedback notifications

There are two important points to note about setting up feedback notifications in multiple Regions:

- Verified identity settings, such as whether you receive feedback by email or through Amazon Simple Notification Service (Amazon SNS), only apply to the Region that you set them in. For example, if you verify `user@example.com` in the US West (Oregon) and US East (N. Virginia) Regions and you want to receive bounced emails via Amazon SNS notifications, you have to use the Amazon SES API or the Amazon SES console to set up Amazon SNS feedback notifications for `user@example.com` in both Regions.
• Amazon SNS topics that you use for feedback forwarding have to be in the same Region where you use Amazon SES.

SMTP credentials

The credentials that you use to send email through the Amazon SES SMTP interface are unique to each AWS Region. If you use the Amazon SES SMTP interface to send email in more than one Region, you have to generate a set of SMTP credentials (p. 86) for each Region.

Note
If you created your SMTP credentials before January 10, 2019, your SMTP credentials were created using an older version of the AWS Signature. For security purposes, you should delete credentials that you created before this date, and replace them with newer credentials. You can delete older credentials by using the IAM console.

Custom MAIL FROM domains

You can use the same custom MAIL FROM domain for verified identities in different AWS Regions. If that is what you want to do, you only need to publish one MX record to the MAIL FROM domain’s DNS server. In this situation, bounce notifications are sent to the Amazon SES feedback endpoint in the Region that you specified in the MX record first. Next Amazon SES redirects the bounces to the verified identity in the Region that sent the email.

Use the MX record settings that Amazon SES provides during the custom MAIL FROM setup process for an identity in one of the Regions. The custom MAIL FROM setup process is described in Setting up a custom MAIL FROM domain (p. 65). For reference, you can find the feedback endpoints for all of the Regions in the following table.

Note
SMTP endpoints are not currently available in Middle East (Bahrain).

<table>
<thead>
<tr>
<th>Region Name</th>
<th>Feedback Endpoints for Custom MAIL FROM Sending Configurations</th>
</tr>
</thead>
<tbody>
<tr>
<td>US East (Ohio)</td>
<td>feedback-smtp.us-east-2.amazonaws.com</td>
</tr>
<tr>
<td>US East (N. Virginia)</td>
<td>feedback-smtp.us-east-1.amazonaws.com</td>
</tr>
<tr>
<td>US West (N. California)</td>
<td>feedback-smtp.us-west-1.amazonaws.com</td>
</tr>
<tr>
<td>US West (Oregon)</td>
<td>feedback-smtp.us-west-2.amazonaws.com</td>
</tr>
<tr>
<td>Asia Pacific (Mumbai)</td>
<td>feedback-smtp.ap-south-1.amazonaws.com</td>
</tr>
<tr>
<td>Asia Pacific (Seoul)</td>
<td>feedback-smtp.ap-northeast-2.amazonaws.com</td>
</tr>
<tr>
<td>Asia Pacific (Singapore)</td>
<td>feedback-smtp.ap-southeast-1.amazonaws.com</td>
</tr>
<tr>
<td>Asia Pacific (Sydney)</td>
<td>feedback-smtp.ap-southeast-2.amazonaws.com</td>
</tr>
<tr>
<td>Asia Pacific (Tokyo)</td>
<td>feedback-smtp.ap-northeast-1.amazonaws.com</td>
</tr>
<tr>
<td>Canada (Central)</td>
<td>feedback-smtp.ca-central-1.amazonaws.com</td>
</tr>
<tr>
<td>Europe (Frankfurt)</td>
<td>feedback-smtp.eu-central-1.amazonaws.com</td>
</tr>
</tbody>
</table>
Sending authorization

Delegate senders can only send emails from the AWS Region where the identity owner’s identity is verified. The sending authorization policy that gives permission to the delegate sender must be attached to the identity in that Region. For more information about sending authorization, see Using sending authorization with Amazon SES (p. 149).

Email receiving

With the exception of Amazon S3 buckets, all of the AWS resources that you use for receiving email with Amazon SES have to be in the same AWS Region as the Amazon SES endpoint. For example, if you use Amazon SES in the US West (Oregon) Region, then any Amazon SNS topics, AWS KMS keys, and Lambda functions that you use also have to be in the US West (Oregon) Region. Similarly, to receive email with Amazon SES within a Region, you have to create an active receipt rule set in that Region.

The following table lists the email receiving endpoints for all of the AWS Regions where Amazon SES supports email receiving:

<table>
<thead>
<tr>
<th>Region Name</th>
<th>Email Receiving Endpoint</th>
</tr>
</thead>
<tbody>
<tr>
<td>US East (N. Virginia)</td>
<td>inbound-smtp.us-east-1.amazonaws.com</td>
</tr>
<tr>
<td>US West (Oregon)</td>
<td>inbound-smtp.us-west-2.amazonaws.com</td>
</tr>
<tr>
<td>Europe (Ireland)</td>
<td>inbound-smtp.eu-west-1.amazonaws.com</td>
</tr>
</tbody>
</table>

**Note**

Amazon SES doesn’t support email receiving in the following Regions: US East (Ohio), US West (N. California), Asia Pacific (Mumbai), Asia Pacific (Seoul), Asia Pacific (Singapore), Asia Pacific (Sydney), Asia Pacific (Tokyo), Canada (Central), Europe (Frankfurt), Europe (London), Europe (Paris), Europe (Stockholm), Middle East (Bahrain), South America (São Paulo), and AWS GovCloud (US).
Service quotas in Amazon SES

The following sections list and describe the quotas that apply to Amazon SES resources and operations. Some quotas can be increased, while others can’t. To determine whether you can request an increase for a quota, refer to the Eligible for Increase column in each section.

Email sending quotas

The following quotas apply to sending email through Amazon SES.

Sending quotas

Note
Quotas are based on the number of recipients, rather than on the number of messages.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Default Quota</th>
<th>Eligible for Increase?</th>
</tr>
</thead>
</table>
| Number of emails that can be sent per 24-hour period | If your account is in the sandbox, you can send up to 200 emails per 24-hour period.  
If your account is out of the sandbox, this number varies based on your specific use case.  
Note  
This value was referred to in the past as your "sending quota." | Yes (p. 146)                        |
| Number of emails that can be sent per second (sending rate) | If your account is in the sandbox, you can send 1 email per second.  
If your account is out of the sandbox, this rate varies based on your specific use case. | Yes (p. 146)                        |

Message quotas

<table>
<thead>
<tr>
<th>Resource</th>
<th>Default Quota</th>
<th>Eligible for Increase?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum message size (including attachments)</td>
<td>10 MB per message (after base64 encoding).</td>
<td>Yes</td>
</tr>
</tbody>
</table>
## Sender and recipient quotas

<table>
<thead>
<tr>
<th>Resource</th>
<th>Default Quota</th>
<th>Eligible for Increase?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of recipients per message</td>
<td>50 recipients per message.</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> A recipient is any &quot;To&quot;, &quot;CC&quot;, or &quot;BCC&quot; address.</td>
<td></td>
</tr>
<tr>
<td>Maximum number of identities that you can verify</td>
<td>10,000 identities per AWS Region.</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> An <em>identity</em> is a domain or email address that you use to send email through Amazon SES.</td>
<td></td>
</tr>
</tbody>
</table>

## Quotas related to event publishing

<table>
<thead>
<tr>
<th>Resource</th>
<th>Default Quota</th>
<th>Eligible for Increase?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of configuration sets</td>
<td>10,000</td>
<td>No</td>
</tr>
<tr>
<td>Maximum length of configuration set name</td>
<td>Configuration set names can contain up to 64 alphanumeric characters. They can also contain hyphens (-) and underscores (_). Names can't contain spaces, accented characters, or any other special characters.</td>
<td>No</td>
</tr>
<tr>
<td>Maximum number of event destinations per configuration set</td>
<td>10</td>
<td>No</td>
</tr>
<tr>
<td>Maximum number of dimensions per CloudWatch event destination</td>
<td>10</td>
<td>No</td>
</tr>
</tbody>
</table>

## Email template quotas

<table>
<thead>
<tr>
<th>Resource</th>
<th>Default Quota</th>
<th>Eligible for Increase?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of email templates in each AWS Region</td>
<td>10,000</td>
<td>No</td>
</tr>
<tr>
<td>Maximum template size</td>
<td>500 KB</td>
<td>No</td>
</tr>
</tbody>
</table>
Quotas related to email receiving

The following table lists the quotas associated with receiving email through Amazon SES.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Default Quota</th>
<th>Eligible for Increase?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of replacement values in each template</td>
<td>Unlimited</td>
<td>N/A</td>
</tr>
<tr>
<td>Maximum number of recipients for each templated email</td>
<td>50 destinations. A destination is any email address on the &quot;To&quot;, &quot;CC&quot;, or &quot;BCC&quot; lines.</td>
<td>No</td>
</tr>
</tbody>
</table>

Note
The number of destinations you can contact in a single call to the API may be limited by your account's maximum sending rate.

General quotas

The following table lists quotas that apply to both sending and receiving email through Amazon SES.

Amazon SES API quotas
<table>
<thead>
<tr>
<th>Resource</th>
<th>Default Quota</th>
<th>Eligible for Increase?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rate at which you can call Amazon SES API actions</td>
<td>All actions (except for SendEmail and SendRawEmail) are throttled at one request per second. For more information about the Amazon SES API, see the Amazon Simple Email Service API Reference.</td>
<td>No</td>
</tr>
</tbody>
</table>
Best practices for sending email using Amazon SES

The way you manage email communications with your customers is referred to as your email program. There are several factors that can lead to the success or failure of your email program; these factors may seem confusing or mysterious at first. However, by understanding how email is delivered, and by following certain best practices, you can increase the chances of your email successfully reaching your customers' inboxes.

Topics
- Email program success metrics (p. 465)
- Tips and best practices (p. 467)

Email program success metrics

There are several metrics that help measure the success of your email program.

This section provides information about the following metrics:
- Bounces (p. 465)
- Complaints (p. 466)
- Message quality (p. 467)

Bounces

A bounce occurs when an email cannot be delivered to the intended recipient. There are two types of bounces: hard bounces and soft bounces. A hard bounce occurs when the email cannot be delivered because of a persistent issue, such as when an email address doesn't exist. A soft bounce occurs when a temporary issue prevents the delivery of an email. Soft bounces can occur when a recipient's inbox is full, or when the receiving server is temporarily unavailable. Amazon SES handles soft bounces by attempting to re-deliver soft bounced emails for a certain period of time.

It's essential that you monitor the number of hard bounces in your email program, and that you remove hard-bouncing email addresses from your recipient lists. When email receivers detect a high rate of hard bounces, they assume that you don't know your recipients well. As a result, a high hard bounce rate can negatively impact the deliverability of your email messages.

The following guidelines can help you avoid bounces and improve your sender reputation:

- Try to keep your hard bounce rate below 5%. The fewer hard bounces in your email program, the more likely ISPs will see your messages as legitimate and valuable. This rate should be considered a reasonable and attainable goal, but isn't a universal rule across all ISPs.
- Never rent or buy email lists. These lists may contain large numbers of invalid addresses, which could cause your hard bounce rates to increase dramatically. Furthermore, these lists could contain spam traps—email addresses specifically used to catch illegitimate senders. If your messages land in a spam trap, your delivery rates and sender reputation could be irrevocably damaged.
- Keep your list up to date. If you haven't emailed your recipients in a long time, try to validate your customers' statuses through some other means (such as website login activity or purchase history).
- If you don't have a method of verifying your customers' statuses, consider sending a win-back email. A typical win-back email mentions that you haven't heard from the customer in a while, and encourages...
the customer to confirm that they still want to receive your email. After sending a win-back email, purge all of the recipients who did not respond from your lists.

When you receive bounces, it's vital that you respond to them appropriately by observing the following rules:

- If an email address hard bounces, immediately remove that address from your lists. Do not attempt to re-send messages to hard-bouncing addresses. Repeated hard bounces add up, and ultimately harm your reputation with the recipient's ISP.
- Make sure that the address you use to receive bounce notifications is able to receive email. For more information about setting up bounce and complaint notifications, see Monitoring Amazon SES email sending using notifications (p. 265).
- If your inbound email comes to you from an ISP, instead of through your own internal servers, an influx of bounce notifications can land in your spam folder or be dropped completely. Ideally, you should not use a hosted email address to receive bounces. If you must, however, then check the spam folder often, and don't mark the bounce messages as spam. In Amazon SES, you can specify the address that bounce notifications are sent to.
- Usually, a bounce provides the address of the mailbox refusing delivery. However, if you need more granular data to map a recipient address to a particular email campaign, include an X-header with a value you can trace back to your internal tracking system. For more information, see Header fields (p. 511).

Complaints

A complaint occurs when an email recipient clicks the "Mark as Spam" (or equivalent) button in their web-based email client. If you accumulate a large number of these complaints, the ISP assumes that you are sending spam. This has a negative impact on your deliverability rate and sender reputation. Some, but not all, ISPs will notify you when a complaint is reported; this is known as a feedback loop. Amazon SES automatically forwards complaints from ISPs that offer feedback loops to you.

The following guidelines can help you avoid complaints and improve your sender reputation:

- Try to keep your complaint rate below 0.1%. The fewer complaints in your email program, the more likely ISPs will see your messages as legitimate and valuable. This rate should be considered a reasonable and attainable goal, but isn't a universal rule across all ISPs.
- If a customer complains about a marketing email, you should immediately stop sending that customer marketing emails. However, if your email program also includes other types of emails (such as notification or transactional emails), it may be acceptable to continue to send those types of messages to the recipient who issued the complaint.
- As with hard bounces, if you have a list that you haven't sent email to in a while, ensure that your recipients understand why they're receiving your messages. We recommend that you send a welcome message reminding them of who you are and why you're contacting them.

When you receive complaints, it's vital that you respond to them appropriately by observing the following rules:

- Make sure that the address you use to receive complaint notifications is able to receive email. For more information about setting up bounce and complaint notifications, see Monitoring Amazon SES email sending using notifications (p. 265).
- Make sure that your complaint notifications aren't being marked as spam by your ISP or mail system.
- Complaint notifications usually contain the body of the email; this is different from bounce notifications, which only include the email headers. However, in complaint notifications, the email address of the individual who issued the complaint is removed. Use custom X-headers or special
identifiers embedded in the email body so that you can identify the email address that issued the complaint. This technique makes it easier to identify addresses that complained so that you can remove them from your recipient lists.

Message quality

Email receivers use content filters to detect certain attributes in your messages to identify whether your message is legitimate. These content filters automatically review the content of your messages to identify common traits of unwanted to malicious messages. Amazon SES uses content filtering technologies to help detect and block messages that contain malware before they are sent.

If an email receiver's content filters determine that your message contains the characteristics of spam or malicious email, your message will most likely be flagged and diverted from recipients' inboxes.

Remember the following when designing your email:

• Modern content filters are intelligent, continuously adapting and changing. They don't rely on a predefined set of rules. Third-party services such as ReturnPath or Litmus can help identify content in your email that may trigger content filters.
• If your email contains links, check the URLs for those links against DNS-based Blackhole Lists (DNSBLs), such as those found at URIBL.com and SURBL.org.
• Avoid using link shorteners. Malicious senders may use link shorteners to hide the actual destination of a link. When ISPs notice that link shortening services—even the most reputable ones—are being used for nefarious purposes, they may deny access to those services altogether. If your email contains a link to a link shortening service that has been added to a deny list, it won't reach your customers' inboxes, and the success of your email campaign suffers.
• Test every link in your email to ensure that it points to the intended page.
• Make sure your website includes Privacy Policy and Terms of Use documents, and that these documents are up to date. It's a good practice to link to these documents from each email you send. Providing links to these documents demonstrates that you have nothing to hide from your customers, which can help build a relationship of trust.
• If you plan to send high-frequency content (such as "daily deals" messages), ensure that the content of your email is different with each deployment. When you send messages with high frequency, you must ensure that those messages are timely and relevant, rather than repetitive and annoying.

Tips and best practices

Even when you have your customers' best interests in mind, you may still encounter situations that impact the deliverability of your messages. The following sections contain recommendations to help ensure that your email communications reach your intended audience.

General recommendations

• Put yourself in your customer's shoes. Ask yourself if the message you are sending is something you would want to receive in your own inbox. If the answer is anything less than an enthusiastic "yes!" then you probably shouldn't send it.
• Some industries have a reputation for poor quality or even malicious email practices. If you are involved in the following industries, you must monitor your reputation very closely and resolve issues immediately:
  • Home mortgage
  • Credit
Domain and "From" address considerations

- Think carefully about the addresses you send email from. The "From" address is one of the first pieces of information your recipients see, and therefore can leave a lasting first impression. Additionally, some ISPs associate your reputation with your "From" address.

- Consider using subdomains for different types of communications. For example, assume you are sending email from the domain example.com, and you plan to send both marketing and transactional messages. Rather than sending all of your messages from example.com, send your marketing messages from a subdomain such as marketing.example.com, and your transactional messages from a subdomain such as orders.example.com. Unique subdomains develop their own reputations. Using subdomains reduces the risk of damage to your reputation if, for example, your marketing communications land in a spam trap or trigger a content filter.

- If you plan to send a large number of messages, don't send those messages from an ISP-based address such as sender@hotmail.com. If an ISP notices a large volume of messages coming from sender@hotmail.com, that email is treated differently than an email that comes from an outbound email sending domain that you own.

- Work with your domain registrar to ensure that the WHOIS information for your domain is accurate. Maintaining an honest and up-to-date WHOIS record demonstrates that you value transparency, and allows users to quickly identify whether or not your domain is legitimate.

- Avoid using a no-reply address, such as no-reply@example.com, as your "From" or "Reply-to" address. Using a no-reply@ email address sends your recipients a clear message: that you aren't offering them a way to contact you, and that you're not interested in their feedback.

Authentication

- Authenticate your domain with SPF (p. 129) and SenderID. These authentication methods confirm to email recipients that each email you send is actually from the domain it claims to be from.

- Sign your outbound mail with DKIM (p. 130). This step confirms to recipients that the content has not been changed in transit between sender and receiver.

- You can test your authentication settings for both SPF and DKIM by sending an email to an ISP-based email address that you own, such as a personal Gmail or Hotmail account, and then viewing the message's headers. The headers indicate whether your attempts to authenticate and sign the message were successful.

Building and maintaining your lists

- Implement a double opt-in strategy. When users sign up to receive email from you, send them a message with a confirmation link, and do not start sending them email until they confirm their address by clicking that link. A double opt-in strategy helps reduce the number of hard bounces resulting from typographical errors.

- When collecting email addresses with a web-based form, perform minimal validation on those addresses upon submission. For example, ensure that the addresses you collect are well-formed (that is, they are in the format recipient@example.com), and that they refer to domains with valid MX records.
• Use caution when allowing user-defined input to be passed to Amazon SES unchecked. Forums registrations and form submissions present unique risks because the content is completely user-generated, and spammers can fill out forms with their own content. It's your responsibility to ensure that you only send email with high-quality content.

• It is highly unlikely that a standard alias (such as postmaster@, abuse@, or noc@) will ever sign up for your email intentionally. Ensure that you are only sending messages to real people who actually want to receive them. This rule is especially true for standard aliases, which are customarily reserved for email watchdogs. These aliases can be maliciously added to your list as a form of sabotage, in order to damage your reputation.

Compliance

• Be aware of the email marketing and anti-spam laws and regulations in the countries and regions you send email to. You're responsible for ensuring that the email you send complies with these laws. This guide doesn't cover these laws, so it's important that you research them. For a list of laws, see Email Spam Legislation by Country on Wikipedia.

• Always consult an attorney to obtain legal advice.
Troubleshoot Amazon SES issues

This section contains the following topics that may help you when you encounter problems:

- For information about domain verification problems that you might encounter, see Email address and domain verification problems (p. 471).
- For solutions to DKIM-related issues, see Troubleshooting DKIM problems in Amazon SES (p. 473).
- For a list of common delivery problems that you might encounter when you send email, along with corrective actions that you can take, see Amazon SES Delivery problems (p. 474).
- For a description of issues recipients may see when they receive an email that was sent through Amazon SES, see Problems with emails received from Amazon SES (p. 475).
- For solutions to problems with bounce, complaint, and delivery notifications, see Amazon SES notification problems (p. 475).
- For a list of errors that can occur when you send an email with Amazon SES, see Amazon SES email sending errors (p. 476).
- For tips on how to increase your email sending speed when you make multiple calls to Amazon SES using either the API or the SMTP interface, see Increasing throughput with Amazon SES (p. 478).
- For solutions to common problems that you might encounter when you use Amazon SES through its Simple Mail Transfer Protocol (SMTP) interface, as well as a list of SMTP response codes that Amazon SES returns, see Amazon SES SMTP issues (p. 479).
- For a list of error codes that are returned by the Amazon SES Query (HTTPS) API, see Error codes returned by the Amazon SES API (p. 413).
- For a description of common issues related to our sending review process, and how to handle them, see Amazon SES Sending review process FAQs (p. 489).
- For a discussion about how DNS-based Blackhole Lists (DNSBLs) affect your sending with Amazon SES, see DNS Blackhole List (DNSBL) FAQs (p. 503).

If you are calling the Amazon SES API directly, see the Amazon Simple Email Service API Reference for the HTTP errors that you might receive.

General Amazon SES issues

The information on this page will explain and help diagnose issues that you may encounter when using Amazon SES.

Changes that I make are not immediately visible

As a service that is accessed through computers in data centers around the world, Amazon SES uses a distributed computing model called eventual consistency. Any change that you make in Amazon SES (or other AWS services) takes time to become visible from all possible endpoints. Some of the delay results from the time it takes to send the data from server to server and from region to region around the world. In the majority of cases, this delay will be no more than a few minutes.

Some areas in which you may notice a delay include:
• Creating and modifying configuration sets – When you create or modify a configuration set (for example, if you associate a dedicated IP pool with an existing configuration set (p. 256)), there may be a brief delay from the time that you create or modify it to the time those changes are active.

• Creating and modifying event destinations – When you create or modify an event destination (for example, to tell Amazon SES to send your email sending data to another AWS service (p. 288)), there may be a delay between the time you created or modified the event destination and the time email sending events actually arrive at the specified destination.

Email address and domain verification problems

To verify an email address or domain with Amazon SES, you initiate the process using either the Amazon SES console or the Amazon SES API. This section contains information that may help resolve issues with the verification process.

Common email verification problems

• The verification email didn't arrive – If you complete the procedures in Verifying email addresses in Amazon SES (p. 47) but you don't receive the verification email within a few minutes, complete the following steps:
  • Check the spam or junk mail folder for the email address you're attempting to verify.
  • Confirm that the address that you're trying to verify is able to receive email. Using a separate email address (such as your personal email address), send a test email to the address that you want to verify.
  • Check the list of verified addresses in the Amazon SES console. Make sure that there aren't any errors in the email address that you're attempting to verify.

Common domain verification problems

If you attempt to verify a domain using the procedure in Verifying domains in Amazon SES (p. 59) and you encounter problems, review the possible causes and solutions below.

• You're attempting to verify a domain that you don't own – You can't verify a domain that you don't own. For example, if you want to send email through Amazon SES from an address on the gmail.com domain, you need to verify that email address specifically (p. 47). You can't verify the entire gmail.com domain.

• Your DNS provider doesn't allow underscores in TXT record names – Some DNS providers don't allow you to include the underscore character in the DNS record names for your domain. If this is true for your provider, you can omit _amazonses from the name of the TXT record.

• Your DNS provider appended the domain name to the end of the TXT record – Some DNS providers automatically append the name of your domain to the attribute name of TXT record. For example, if you create a record where the attribute name is _amazonses.example.com, the provider might append the domain name, resulting in _amazonses.example.com.example.com. To avoid duplication of the domain name, add a period to the end of the domain name when you create the TXT record. This step tells your DNS provider that it isn't necessary to append the domain name to the TXT record.

• Your DNS provider modified the DNS record value – Some providers automatically modify DNS record values to use only lowercase letters. Amazon SES only verifies your domain when it detects a verification record for which the attribute value exactly matches the value that Amazon SES provided when you started the domain verification process. If the DNS provider for your domain changes your TXT record values to use only lowercase letters, contact the DNS provider for additional assistance.

• You want to verify the same domain multiple times – You might need to verify your domain more than once because you're sending in different regions, or because you're using the same domain to send from multiple AWS accounts. If your DNS provider doesn't allow you to have more than one
TXT record with the same attribute name, you might still be able to verify two domains. If your DNS provider allows it, you can assign multiple attribute values to the same TXT record. For example, if your DNS is managed by Amazon Route 53, you can set up multiple values for the same TXT record by completing the following steps:

1. In the Route 53 console, choose the TXT record you created when you verified your domain in the first region.
2. In the Value box, go to the end of the existing attribute value, and then press Enter.
3. Add the attribute value for the additional region, and then save the record set.

If your DNS provider doesn't let you assign multiple values to the same TXT record, you can verify the domain once with _amazonses in the attribute name of the TXT record, and another time with _amazonses removed from the attribute name. The downside of this solution is that you can only verify the same domain two times.

Checking domain verification settings

You can check that your Amazon SES domain verification TXT record is published correctly to your DNS server by using the following procedure. This procedure uses the nslookup tool, which is available for Windows and Linux. On Linux, you can also use dig.

The commands in these instructions were executed on Windows 7, and the example domain we use is ses-example.com.

In this procedure, you first find the DNS servers that serve your domain, and then query those servers to view the TXT records. You query the DNS servers that serve your domain because those servers contain the most up-to-date information for your domain, which can take time to propagate to other DNS servers.

To verify that your domain verification TXT record is published to your DNS server

1. Find the name servers for your domain by taking the following steps.
   a. Go to the command line. To get to the command line on Windows 7, choose Start and then type cmd. On Linux-based operating systems, open a terminal window.
   b. At the command prompt, type the following, where <domain> is your domain. This will list all of the name servers that serve your domain.

   ```
   nslookup -type=NS <domain>
   ```

   If your domain was ses-example.com, this command would look like:

   ```
   nslookup -type=NS ses-example.com
   ```

   The command's output will list the name servers that serve your domain. You will query one of these servers in the next step.

2. Verify that the TXT record is correctly published by taking the following steps.
   a. At the command prompt, type the following, where <domain> is your domain, and <name server> is one of the name servers you found in step 1.

   ```
   nslookup -type=TXT _amazonses.<domain> <name server>
   ```

   In our ses-example.com example, if a name server that we found in step 1 was called ns1.nameserver.net, we would type the following:
nslookup -type=TXT _amazonses.ses-example.com ns1.name-server.net

b. In the output of the command, verify that the string that follows `text =` matches the TXT value you see when you choose the domain in the Identities list of the Amazon SES console.

In our example, we are looking for a TXT record under `_amazonses.ses-example.com` with a value of `fmxqxT/icOYx4aA/bEUrDPMeax9/s3frblS+niixmqk=`. If the record is correctly published, we would expect the command to have the following output:

```plaintext
_amazonses.ses-example.com text = "fmxqxT/icOYx4aA/bEUrDPMeax9/s3frblS+niixmqk="
```

Troubleshooting DKIM problems in Amazon SES

This section lists some of the problems that you may encounter when you configure DKIM authentication in Amazon SES. If you attempt to set up DKIM and you encounter problems, review the possible causes and solutions below.

You set up DKIM successfully, but your messages aren't being DKIM-signed

If you used Easy DKIM (p. 131) or BYODKIM (p. 138) to configure DKIM for a domain, but the messages that you send aren't DKIM-signed, do the following:

- Make sure that DKIM is enabled for the appropriate identity. To enable DKIM for an identity in the Amazon SES console, choose the email domain in the Identities list. On the details page for the domain, expand DKIM, and then choose Enable to enable DKIM.

- Make sure that you're not sending from a verified email address on the same domain. If you set up DKIM for a domain, then all of the messages that you send from that domain are DKIM-signed, except for email addresses that you verified individually. Individually verified email addresses use separate settings. For example, if you configured DKIM for the domain `example.com`, and you separately verified the email address `mary@example.com` (but didn't configure DKIM for the address), then emails that you send from `mary@example.com` are sent without DKIM authentication. You can resolve this issue by deleting the email address identity from the list of identities for your account.

- If you use the same identity in more than one AWS Region, you have to configure DKIM for each region separately. Similarly, if you use the same domain with more than one AWS account, you have to configure DKIM for each account. If you remove the necessary DNS records for a specific region or account, Amazon SES disables DKIM signing in that region or account. If DKIM signing becomes disabled, Amazon SES sends you a notification by email.

Your domain's DKIM details in the Amazon SES console show DKIM: waiting on sender verification...

DKIM Verification Status: pending verification.

If you complete the procedures in Easy DKIM (p. 131) or Provide Your Own DKIM Authentication Token (p. 138) to configure DKIM for a domain, but the Amazon SES console still indicates that DKIM verification is pending, do the following:

- Wait up to 72 hours. In rare cases, it can take time for the DNS records to become visible to Amazon SES.

- Confirm that the CNAME record (for Easy DKIM) or the TXT record (for BYODKIM) uses the correct name. Some DNS providers automatically append the domain name to records that you create. For example, if you create a record with a Name of `example._domainkey.example.com`, your DNS provider might add the name of your domain to the end of this string, resulting in `example._domainkey.example.com.example.com`. For more information, see the documentation for your DNS provider.
You receive an email from Amazon SES that says your DKIM setup has been (or will be) revoked.

This means that Amazon SES can no longer find the required CNAME records (if you used Easy DKIM) or the required TXT record (if you used BYODKIM) records on your DNS server. The notification email will inform you of the length of time in which you must re-publish the DNS records before your DKIM setup status is revoked and DKIM signing is disabled. If your DKIM setup is revoked, you must restart the DKIM set-up procedure from the beginning.

When attempting to set up BYODKIM, the DKIM verification process fails.

Make sure that your private key uses the right format. The private key has to be in PKCS #1 format and use 1024-bit RSA encryption. Additionally, the private key has to be base64 encoded.

While setting up BYODKIM, you receive a BadRequestException error when you try to specify a public key for the domain.

If you receive a BadRequestException error, do the following:

- Make sure that the selector that you specify for the public key contains at least 1 and less than or equal to 63 alphanumeric characters. The selector can't include periods or other symbols or punctuation.
- Make sure that you've removed the header and footer lines from the public key, and that you've removed all of the line breaks from the public key.

When using Easy DKIM, your DNS servers successfully return the Amazon SES DKIM CNAME records, but return SERVFAIL for the domain verification TXT record.

Your DNS provider might not be able to redirect CNAME records. Amazon SES and ISPs query for TXT records. To comply with the DKIM specification, your DNS servers have to be able to respond to TXT record queries as well as CNAME record queries. If your DNS provider isn't able to respond to TXT record queries, an alternative is to use Route 53 as your DNS hosting provider.

Your emails contain two DKIM signatures

The extra DKIM signature, which contains d=amazonses.com, is automatically added by Amazon SES. You can ignore it.

Amazon SES Delivery problems

After you make a successful request to Amazon SES, your message is often sent immediately. At other times, there might be a short delay. In any case, you can be assured that your email will be sent.

When Amazon SES sends your message, however, several factors can prevent it from being delivered successfully, and in some cases you will become aware that delivery failed only when the message you send does not arrive. Use the following process to resolve this situation.

If an email does not arrive, try the following:

- Verify that you made a SendEmail or SendRawEmail request for the email in question and that you received a successful response. If you are making these requests programmatically, check your software logs to ensure that the program made the request and received a successful response.
- Read the blog article Three places where your email could get delayed when sending through SES because the problem might actually be a delay rather than a nondelivery.
- Check the sender's email address (the "From" address) to verify that it is valid. Also check the Return-Path address, which is where bounce messages are sent. If your mail bounced, there will be an explanatory error message there.
- Check the AWS Service Health Dashboard to confirm that there is not a known problem with Amazon SES.
Problems with received emails

• Contact the email recipient or the recipient's ISP. Verify that the recipient is using the correct email address, and inquire whether there have been any known delivery problems with the recipient's ISP. Also, determine whether the email did arrive but was filtered as spam.

• If you have signed up for a paid AWS Support Plan, you can open a new technical support case. In your correspondence with us, please provide any relevant recipient addresses, along with any request IDs or message IDs returned from the SendEmail or SendRawEmail responses.

• Wait to see if the problem is actually a delay, not a permanent delivery failure. To combat spammers, some ISPs temporarily reject incoming messages from unknown sending mail servers. This process, called greylisting, can cause a delay in delivery. Amazon SES will retry these messages. If greylisting is the issue, the ISP might accept the email on one of these retry attempts.

Problems with emails received from Amazon SES

This section discusses some common issues that you might see when you receive emails that were sent from Amazon SES.

The email client displays "sent via amazonses.com" as the source of the email

Some email clients display the "via" domain when the sender's domain doesn't match the domain that the email was sent from (in this case, amazonses.com). For more information, see Extra info next to sender's name on the Gmail Support website. Alternatively, you can set up DomainKeys Identified Mail (DKIM). When you authenticate your emails using DKIM, email clients typically don't show the "via" domain because the DKIM signature shows that the email is from the domain that it claims to be from. For information about setting up DKIM, see Authenticating Email with DKIM in Amazon SES (p. 130).

The message contains garbled or nonsense characters

If your message includes characters that aren't in the ASCII character set (such as accented Latin characters, Chinese characters, or Arabic characters), you have to encode those characters using HTML character encoding. You can use web-based tools to encode the characters in your email, such as the HTML Character Converter on the Email On Acid website.

Alternatively, you can assemble the MIME message yourself. In the MIME message, you can specify that the message should use UTF-8 encoding. When you use UTF-8 encoding, you can use non-ASCII characters directly in your messages. When you've finished creating the MIME message, you can send it using the SendRawEmail API. To learn more about sending messages using the SendRawEmail API, see Sending Raw Email using AWS SDKs (p. 442).

One common cause of this issue is the Smart Quotes feature of Microsoft Word. If you often copy content from Word and paste it into your emails, you might encounter this issue. The Smart Quotes feature replaces straight quote characters ("...") with curly quote characters ("..."). Curly quote characters aren't standard ASCII characters. As a result, they might be rendered in some email clients as "??" or as a group of characters such as "&eacute;". To correct this issue, you can disable the Smart Quotes feature in Word. Alternatively, you can use the SendRawEmail solution from the preceding paragraph. To learn how to disable this feature, see Smart quotes in Word on the Microsoft Office Support website.

Amazon SES notification problems

If you encounter a problem with bounce, complaint, or delivery notifications, review the possible causes and solutions below.

• You receive bounce notifications via Amazon SNS, but you don't know which recipients the notifications correspond to—In the future, to associate a bounce notification with a given recipient, you have the following options:
• Since Amazon SES doesn't retain any custom message IDs that you have added, store a mapping between an identifier and the Amazon SES message ID that Amazon SES passes back to you when it accepts the email.
• In each call to Amazon SES, send to a single recipient, rather than sending a single message to multiple recipients.
• You can enable feedback forwarding via email, which will forward the full bounce message to you.

You receive complaint notifications via Amazon SNS or email feedback forwarding, but you don't know which recipients the notifications correspond to—Some ISPs redact the complained recipient’s email address before passing the complaint notification to Amazon SES. To enable you to find the recipient’s email address, your best option is to store your own mapping between an identifier and the Amazon SES message ID that Amazon SES passes back to you when it accepts the email. Note that Amazon SES does not retain any custom message IDs that you add.

You want to set up notifications to go to an Amazon SNS topic you don’t own—The owner of that topic must configure an Amazon SNS access policy that allows your account to call the SNS:Publish action on their topic. For information about how to control access to your Amazon SNS topic through the use of IAM policies, see Managing Access to Your Amazon SNS Topics in the Amazon Simple Notification Service Developer Guide.

Amazon SES email sending errors

This topic reviews the types of email sending-specific errors that you may encounter when you send an email through Amazon SES. If you try to send an email through Amazon SES and the call to Amazon SES fails, Amazon SES returns an error message to your application and does not send the email. The way that you observe this error message depends on the way that you call Amazon SES.

• If you call the Amazon SES API directly, the Query action will return an error. The error may be MessageRejected or one of the errors specified in the Common Errors topic of the Amazon Simple Email Service API Reference.
• If you call Amazon SES using an AWS SDK that uses a programming language that supports exceptions, Amazon SES may throw an exception. The type of exception depends on the SDK and on the error. For example, the exception could be an Amazon SES MessageRejectedException (the actual name may vary depending on the SDK) or a general AWS exception. Regardless of the type of exception, the error type and the error message in the exception will give you more information.
• If you call Amazon SES through its SMTP interface, the way that you experience the error depends on the application. Some applications might display a specific error message, and others might not. For a list of SMTP response codes that Amazon SES returns, see SMTP response codes returned by Amazon SES (p. 480).

Note
When your call to Amazon SES to send an email fails, you are not billed for that email.

The following are the types of Amazon SES-specific problems that can cause Amazon SES to return an error when you try to send an email. These errors are in addition to general AWS errors like MalformedQueryString as specified in the Common Errors topic of the Amazon Simple Email Service API Reference.

• Email address is not verified. The following identities failed the check in region region: identity1, identity2, identity3—You are trying to send email from an email address or domain that you have not verified with Amazon SES (p. 47). This error could apply to the “From”, “Source”, “Sender”, or “Return-Path” address. If your account is still in the Amazon SES sandbox (p. 71), you also must verify every recipient email address except for the recipients provided by the Amazon SES mailbox simulator (p. 181). If Amazon SES is not able to show all of the failed identities, the error message ends with an ellipsis.

476
Note
Amazon SES has endpoints in multiple AWS Regions (p. 457), and email address verification status is separate for each AWS Region. You must complete the verification process for each sender in the AWS Region(s) you want to use.

- **Account is paused**—Your account's ability to send email is paused. You can still access the Amazon SES console and perform most operations. However, if you try to send an email, you receive this message.

  If we pause your account's ability to send email, we automatically send a notification to the email address associated with your AWS account. For more information, see the section called “Sending review process” (p. 489).

- **Throttling**—Your application may be trying to send too many messages per second, or you may have sent too much email over the last 24 hours. In these cases, the error message may be similar to the following examples:
  - **Daily message quota exceeded**—You have sent the maximum number of messages that you are permitted in a 24-hour period. If you have exceeded your daily quota, you will have to wait until the next 24-hour period before you can send more email.
  - **Maximum sending rate exceeded**—You are attempting to send more emails per second than is permitted by your maximum send rate. If you have exceeded your sending rate, you can continue to send email, but will need to reduce your send rate. For more information, see How to handle a “Throttling - Maximum sending rate exceeded” error on the AWS Messaging and Targeting Blog.
  - **Maximum SigV2 SMTP sending rate exceeded**—You are attempting to send messages using SMTP credentials created before January 10, 2019; your SMTP credentials were created using an older version of the AWS Signature. For security purposes, you should delete credentials that you created before this date, and replace them with newer credentials. You can delete older credentials by using the IAM console. For more information, see the section called “Obtaining your SMTP credentials” (p. 86) for creating credentials.

You should regularly monitor your sending activity to see how close you are to your sending quotas. For more information, see Monitoring your Amazon SES sending quotas (p. 145). For general information about sending quotas, see Managing your Amazon SES sending quotas (p. 144). For information about how to increase your sending quotas, see Increasing your Amazon SES sending quotas (p. 146).

  **Important**
  If the error text that explains the throttling error is not related to you exceeding your daily quota or maximum send rate, then there might be a system-wide problem that is causing reduced sending capabilities. For information about the service status, go to the AWS Service Health Dashboard.

- **There are no recipients specified**—No recipients were provided.

- **There are non-ASCII characters in the email address**—The email address string must be 7-bit ASCII. If you want to send to or from email addresses that contain Unicode characters in the domain part of an address, you must encode the domain using Punycode. Punycode is not permitted in the local part of the email address (the part before the @ sign) nor in the “friendly from” name. If you want to use Unicode characters in the “friendly from” name, you must encode the “friendly from” name using MIME encoded-word syntax, as described in Sending raw email using the Amazon SES API (p. 110). For more information about Punycode, see RFC 3492.

- **Mail FROM domain is not verified**—Amazon SES could not read the MX record required to use the specified MAIL FROM domain. For information setting up custom MAIL FROM domains, see Setting up a custom MAIL FROM domain (p. 65).

- **Configuration set does not exist**—The configuration set that you specified does not exist. A configuration set is an optional parameter that you use to publish email sending events. For more information, see Monitor email sending using Amazon SES event publishing (p. 288).
Increasing throughput with Amazon SES

When you send emails, you can call Amazon SES as frequently as your maximum send rate allows. (For more information about your maximum send rate, see Managing your Amazon SES sending quotas (p. 144).) However, each call to Amazon SES takes time to complete.

If you are making multiple calls to Amazon SES using the Amazon SES API or the SMTP interface, you may want to consider the following tips to help you improve your throughput:

• **Measure your current performance to identify bottlenecks**—A possible performance test involves sending multiple test emails as quickly as possible within a code loop in your application. Measure the round-trip latency of each `SendEmail` request. Then, incrementally launch additional instances of the application on the same machine, and watch for any impact on network latency. You may also want to run this test on multiple machines and on different networks to help pinpoint any possible machine resource bottlenecks or network bottleneck that may exist.

• **(API only) Consider using persistent HTTP connections**—Rather than incurring the overhead of establishing a separate new HTTP connection for each API request, use persistent HTTP connections. That is, reuse the same HTTP connection for multiple API requests.

• **Consider using multiple threads**—When an application uses a single thread, the application code calls the Amazon SES API and then synchronously waits for an API response. Sending emails is typically an I/O-bound operation, and doing the work from multiple threads provides better throughput. You can send concurrently using as many threads of execution as you wish.

• **Consider using multiple processes**—Using multiple processes can help increase your throughput because you will have more concurrent active connections to Amazon SES. For example, you can segment your intended emails into multiple buckets, and then run multiple instances of your email sending script simultaneously.

• **Consider using a local mail relay**—Your application can quickly transmit messages to your local mail server, which can then help to buffer the messages and asynchronously transmit them to Amazon SES. Some mail servers support delivery concurrency, which means that even if your application is generating emails to the mail server in a single-threaded fashion, the mail server will use multiple threads when sending to Amazon SES. For more information, see Integrating Amazon SES with your existing email server (p. 93).

• **Consider hosting your application closer to the Amazon SES API endpoint**—You may wish to consider hosting your application in a data center close to the Amazon SES API endpoint, or on an Amazon EC2 instance in the same AWS Region as the Amazon SES API endpoint. This can help to decrease network latency between your application and Amazon SES, and improve throughput. For a list of regions where Amazon SES is available, see Amazon Simple Email Service (Amazon SES) in the AWS General Reference.

• **Consider using multiple machines**—Depending on the system configuration on your host machine, there may be a limit on the number of simultaneous HTTP connections to a single IP address, which may limit the benefits of parallelism once you exceed a certain number of concurrent connections on a single machine. If this is a bottleneck, you may wish to consider making concurrent Amazon SES requests using multiple machines.

• **Consider using the Amazon SES query API instead of the SMTP endpoint**—Using the Amazon SES query API enables you to submit the email send request using a single network call, whereas interfacing with the SMTP endpoint involves an SMTP conversation which consists of multiple network requests (for example, EHLO, MAIL FROM, RCPT TO, DATA, QUIT). For more information about the Amazon SES query API, see Using the Amazon SES API to send email (p. 109).

• **Use the Amazon SES mailbox simulator to test your maximum throughput**—To test any changes you may implement, you can use the mailbox simulator. The mailbox simulator can help you to determine your system’s maximum throughput without using up your daily sending quota. For information about the mailbox simulator, see Testing email sending in Amazon SES (p. 181).
If you are accessing Amazon SES through its SMTP interface, see Amazon SES SMTP issues (p. 479) for specific SMTP-related issues that may affect throughput.

Amazon SES SMTP issues

This section contains solutions for several common issues related to sending email through the Amazon SES Simple Mail Transfer Protocol (SMTP) interface. It also contains a list of SMTP response codes that Amazon SES returns.

To learn more about sending email through the Amazon SES SMTP interface, see Using the Amazon SES SMTP interface to send email (p. 85).

• You can't connect to the Amazon SES SMTP endpoint.

Problems connecting to the Amazon SES SMTP endpoint are most commonly related to the following issues:

• Incorrect credentials – The credentials that you use to connect to the SMTP endpoint are different from your AWS credentials. To obtain your SMTP credentials, see Obtaining your Amazon SES SMTP credentials (p. 86). For more information about credentials, see Types of Amazon SES credentials (p. 16).

• Network or firewall issues – Your network might be blocking outbound connections over the port you're trying to send email from. To determine if an issue on your local network is causing connection issues, type the following command at the command line, replacing \texttt{port} with the port you're trying to use (typically 465, 587, 2465, or 2587): \texttt{telnet email-smtp.us-west-2.amazonaws.com port}

If you are able to connect to the SMTP server using this command, and you are trying to connect to Amazon SES using TLS Wrapper or STARTTLS, complete the procedures shown in Test your connection to the Amazon SES SMTP interface using the command line (p. 103).

If you can't connect to the Amazon SES SMTP endpoint using \texttt{telnet} or \texttt{openssl}, it indicates that something in your network (such as a firewall) is blocking outbound connections over the port you're trying to use. Work with your network administrator to diagnose and fix the problem.

• You're sending to Amazon SES from an Amazon EC2 instance using port 25, and you're receiving timeout errors.

Amazon EC2 restricts port 25 by default. To remove these restrictions, submit an Amazon EC2 Request to Remove Email Sending Limitations. You can also connect to Amazon SES using ports 465 or 587, neither of which is restricted.

• Network errors are causing dropped emails.

Ensure that your application uses retry logic when it connects to the Amazon SES SMTP endpoint, and that your application can detect and retry message delivery in case of a network error. SMTP is a verbose protocol, and sending an email using this protocol requires several network round trips. Because of the nature of SMTP, the potential for network errors increases.

• You lose connection with the SMTP endpoint.

Lost connections are most commonly caused by the following issues:

• MTU size – If you receive a time-out error message, the Maximum Transmission Unit (MTU) of the network interface for the computer you're using to connect to the Amazon SES SMTP interface may be too large. To resolve this issue, set the MTU size on that computer to 1500 bytes.

For more information about setting the MTU size on Windows, Linux, and macOS operating systems, see Queries Appear to Hang in the Client and Do Not Reach the Cluster in the Amazon Redshift Cluster Management Guide.
For more information about setting the MTU size for an Amazon EC2 instance, see Network Maximum Transmission Unit (MTU) for Your EC2 Instance in the Amazon EC2 User Guide for Linux Instances.

- **Long-lived connections** – The Amazon SES SMTP endpoint runs on a fleet of Amazon EC2 instances behind an Elastic Load Balancer (ELB). In order to ensure that the system is up-to-date and fault tolerant, active Amazon EC2 instances are periodically terminated and replaced with new instances. Because your application connects to an Amazon EC2 instance through the ELB, the connection becomes invalid when the Amazon EC2 instance is terminated. You should establish a new SMTP connection after you have delivered a fixed number of messages via a single SMTP connection, or if the SMTP connection has been active for some amount of time. You will need to experiment to find appropriate thresholds depending on where your application is hosted and how it submits email to Amazon SES.

- **You want to know the IP addresses of the Amazon SES SMTP mail servers so that you can whitelist the IP addresses with your network.**

The IP addresses for the Amazon SES SMTP endpoints reside behind load balancers. As a result, these IP addresses change frequently. It’s not possible to provide a definitive list of all of the IP addresses for the Amazon SES endpoints. We recommend that you whitelist the amazonses.com domain, rather than whitelisting individual IP addresses.

### SMTP response codes returned by Amazon SES

This section contains a list of response codes that the Amazon SES SMTP interface returns.

You should retry SMTP requests that receive 400 errors. We recommend that you implement a system that retries requests with progressively longer wait times (for example, wait 5 seconds before retrying, then wait 10 seconds, and then wait 30 seconds). If the third retry doesn’t succeed, wait 20 minutes, and then repeat the process. To see an example of an implementation that uses an exponential retry policy, see How to handle a “Throttling - Maximum sending rate exceeded” error on the AWS Messaging and Targeting Blog.

**Note**

AWS SDKs implement retry logic automatically, but they use the HTTPS interface instead of SMTP.

If you receive a 500 error, you have to revise your request to correct an issue before you submit the request again. For example, if your AWS authentication credentials are invalid, you have to update your application to use the correct credentials before you submit your request again.

<table>
<thead>
<tr>
<th>Description</th>
<th>Response code</th>
<th>More information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication successful</td>
<td>235</td>
<td>Authentication successful</td>
</tr>
<tr>
<td></td>
<td>235 Authentication successful</td>
<td>Your SMTP client successfully connected and signed in to the SMTP server.</td>
</tr>
<tr>
<td>Successful delivery</td>
<td>250</td>
<td>Ok MessageID</td>
</tr>
<tr>
<td></td>
<td>250 Ok MessageID</td>
<td>MessageID is a unique string of characters that Amazon SES uses to identify a message.</td>
</tr>
<tr>
<td>Service unavailable</td>
<td>421</td>
<td>Too many concurrent SMTP connections</td>
</tr>
<tr>
<td></td>
<td>421 Too many concurrent SMTP connections</td>
<td>Amazon SES can't process the request because there are currently too many connections to the SMTP server.</td>
</tr>
<tr>
<td>Local processing error</td>
<td>451</td>
<td>Temporary service failure</td>
</tr>
<tr>
<td></td>
<td>451 Temporary service failure</td>
<td>Amazon SES couldn't process the request. There might be issues.</td>
</tr>
<tr>
<td>Description</td>
<td>Response code</td>
<td>More information</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>--------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Timeout</td>
<td>451 Timeout waiting for data from client</td>
<td>Too much time elapsed between requests, so the SMTP server closed the connection.</td>
</tr>
<tr>
<td>Daily sending quota exceeded</td>
<td>454 Throttling failure: Daily message quota exceeded</td>
<td>You've exceeded the maximum number of emails that Amazon SES permits you to send in a 24-hour period. For more information, see Managing your Amazon SES sending quotas (p. 144).</td>
</tr>
<tr>
<td>Maximum send rate exceeded</td>
<td>454 Throttling failure: Maximum sending rate exceeded</td>
<td>You've exceeded the maximum number of emails that Amazon SES permits you to send per second. For more information, see Managing your Amazon SES sending quotas (p. 144).</td>
</tr>
<tr>
<td>Amazon SES issue when validating SMTP credentials</td>
<td>454 Temporary authentication failure</td>
<td>Issues that could cause this issue include (but aren't limited to):</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• There is a problem with the encryption between your email-sending application and Amazon SES. Note that you have to use an encrypted connection when you connect to Amazon SES. For more information, see Connecting to an Amazon SES SMTP endpoint (p. 90).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Amazon SES could be experiencing an issue. Check the AWS Service Health Dashboard for updates.</td>
</tr>
<tr>
<td>Problem receiving the request</td>
<td>454 Temporary service failure</td>
<td>Amazon SES didn't successfully receive the request. As a result, the message wasn't sent.</td>
</tr>
<tr>
<td>Incorrect credentials</td>
<td>530 Authentication required</td>
<td>The application that you use to send email didn't attempt to authenticate when it connected to the Amazon SES SMTP interface.</td>
</tr>
<tr>
<td>Authentication Credentials Invalid</td>
<td>535 Authentication Credentials Invalid</td>
<td>The application that you use to send email didn't provide the correct SMTP credentials to Amazon SES. Note that your SMTP credentials aren’t the same as your AWS credentials. For more information, see Obtaining your Amazon SES SMTP credentials (p. 86).</td>
</tr>
<tr>
<td>Description</td>
<td>Response code</td>
<td>More information</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Account not subscribed to Amazon SES</td>
<td>535 Account not subscribed to SES</td>
<td>The AWS account that owns the SMTP credentials is not signed up for Amazon SES.</td>
</tr>
<tr>
<td>Message is too long</td>
<td>552 Message is too long.</td>
<td>The message that you're trying to send is larger than 10 MB in size.</td>
</tr>
<tr>
<td>Account not subscribed to Amazon SES</td>
<td>535 Account not subscribed to SES</td>
<td>The AWS account that owns the SMTP credentials is not signed up for Amazon SES.</td>
</tr>
<tr>
<td>User not authorized to call the Amazon SES SMTP endpoint</td>
<td>554 Access denied: User UserARN is not authorized to perform ses:SendRawEmail on resource IdentityARN</td>
<td>The AWS Identity and Access Management (IAM) policy or the Amazon SES sending authorization policy of the user who owns the SMTP credentials isn't allowed to call the Amazon SES SMTP endpoint.</td>
</tr>
<tr>
<td>Unverified email address</td>
<td>554 Message rejected: Email address is not verified. The following identities failed the check in region region: identity0, identity1, identity2</td>
<td>You're trying to send email from an email address or domain that isn't verified to send email from your Amazon SES account (p. 47). This error could apply to the &quot;From&quot;, &quot;Source&quot;, &quot;Sender&quot;, or &quot;Return-Path&quot; addresses. If your account is still in the sandbox, you also have to verify every recipient email address (except for the recipients provided by the Amazon SES mailbox simulator (p. 181)). If Amazon SES isn't able to show all of the identities that failed the verification check, the error message ends with three periods (...). <strong>Note</strong> Amazon SES has endpoints in several AWS Regions (p. 457), and email address verification status is separate for each AWS Region. You have to complete the verification process for each sender in the AWS Regions that you want to use.</td>
</tr>
</tbody>
</table>
Deleting personal data from Amazon SES

Depending on how you use it, Amazon SES might store certain data that could be considered personal. For example, in order to send email using Amazon SES, you must provide at least one verified identity (an email address or a domain). You can use the Amazon SES console or the Amazon SES API to permanently delete this personal data.

This chapter provides procedures for deleting various types of data that might be considered personal.

Topics in this section:
- Delete Email Addresses From the Account-Level Suppression List (p. 483)
- Delete Data About Email Sent Using Amazon SES (p. 484)
- Delete Data About Identities (p. 485)
- Delete Sender Authentication Data (p. 485)
- Delete Data Related to Receiving Rules (p. 486)
- Delete Data Related to IP Address Filters (p. 486)
- Delete Data in Email Templates (p. 487)
- Delete Data in Custom Verification Email Templates (p. 487)
- Delete All Personal Data by Closing Your AWS Account (p. 488)

Delete Email Addresses From the Account-Level Suppression List

Amazon SES includes an optional account-level suppression list. When you enable this feature, email addresses are automatically added to a suppression list when they result in a bounce or complaint. Email addresses remain on this list until you delete them. For more information about the account-level suppression list, see Using the account-level suppression list (p. 184).

You can remove email addresses from the account-level suppression list by using the DeleteSuppressedDestination operation in the Amazon SES API v2. This section includes a procedure for deleting email addresses by using the AWS CLI. For more information about installing and configuring the AWS CLI, see the AWS Command Line Interface User Guide.

To remove an address from the account-level suppression list by using the AWS CLI

At the command line, enter the following command:

```bash
aws sesv2 delete-suppressed-destination --email-address recipient@example.com
```

In the preceding command, replace `recipient@example.com` with the email address that you want to remove from the account-level suppression list.
Delete Data About Email Sent Using Amazon SES

When you use Amazon SES to send an email, you can send information about that email to other AWS services. For example, you can send information about email events (such as deliveries, opens, and clicks) to Kinesis Data Firehose. This event data typically contains your email address and the IP address the email was sent from. It also contains the email addresses of all the recipients the email was sent to.

You can use Kinesis Data Firehose to stream email event data to several destinations—including Amazon Simple Storage Service, Amazon OpenSearch Service, and Amazon Redshift. To remove this data, you should first stop streaming data to Kinesis Data Firehose, and then delete the data that has already been streamed. To stop streaming Amazon SES event data to Kinesis Data Firehose, you must delete the Kinesis Data Firehose event destination.

To remove a Kinesis Data Firehose event destination by using the Amazon SES console

1. Open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. Under Email Sending, choose Configuration Sets.
3. In the list of configuration sets, choose the configuration set that contains the Kinesis Data Firehose event destination.
4. Next to the Kinesis Data Firehose event destination that you want to delete, choose the delete button.
5. If necessary, remove the data that Kinesis Data Firehose wrote to other services. For more information, see the section called “Remove Stored Event Data” (p. 484).

You can also use the Amazon SES API to delete event destinations. The following procedure uses the AWS Command Line Interface (AWS CLI) to interact with the Amazon SES API. You can also interact with the API by using an AWS SDK, or by making HTTP requests directly.

To remove a Kinesis Data Firehose event destination by using the AWS CLI

1. At the command line, type the following command:

   ```bash
   aws ses delete-configuration-set-event-destination --configuration-set-name configSet --event-destination-name eventDestination
   ```

   In this command, replace `configSet` with the name of the configuration set that contains the Kinesis Data Firehose event destination. Replace `eventDestination` with the name of the Kinesis Data Firehose event destination.

2. If necessary, remove the data that Kinesis Data Firehose wrote to other services. For more information, see the section called “Remove Stored Event Data” (p. 484).

Remove Stored Event Data

For more information about deleting information from other AWS services, see the following documents:

- Delete an Object and Bucket in the Amazon Simple Storage Service Getting Started Guide
- Delete an OpenSearch Service Domain in the Amazon OpenSearch Service Developer Guide
- Deleting a Cluster in the Amazon Redshift Cluster Management Guide

You can also use Kinesis Data Firehose to stream email data to Splunk, a third-party service that isn't supported by AWS or managed in the AWS Management Console. For more information about removing data from Splunk, consult your system administrator or the documentation on the Splunk website.
Delete Data About Identities

Identities include the email addresses and domains that you use to send email using Amazon SES. In some jurisdictions, email addresses or domains might be considered personally identifiable data.

To delete an identity by using the Amazon SES console

1. Open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. Under Identity Management, do one of the following:
   - Choose Domains if you want to delete a domain.
   - Choose Email Addresses if you want to delete an email address.
3. Choose the identity that you want to delete, and then choose Remove.
4. On the confirmation dialog box, choose Yes, Delete Identity.

You can also use the Amazon SES API to delete identities. The following procedure uses the AWS Command Line Interface (AWS CLI) to interact with the Amazon SES API. You can also interact with the API by using an AWS SDK, or by making HTTP requests directly.

To delete an identity by using the AWS CLI

• At the command line, type the following command:

   aws ses delete-identity --identity sender@example.com

   In this command, replace sender@example.com with the identity that you want to delete.

Delete Sender Authentication Data

Sender authentication refers to the process of configuring Amazon SES so that another user can send email on your behalf. To enable sender authorization, you must create a policy, as described in Using sending authorization with Amazon SES (p. 149). These policies contain identities (which belong to you), in addition to AWS IDs (which are associated with the person or group that sends email on your behalf). You can remove this personal data by modifying or deleting the sender authentication policies. The following procedures show you how to delete these policies.

To delete a sender authentication policy by using the Amazon SES console

1. Open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. Under Identity Management, do one of the following:
   - Choose Domains if the sender authentication policy you want to delete is associated with a domain.
   - Choose Email Addresses if the sender authentication policy you want to delete is associated with an email address.
3. Under Identity Policies, choose the policy you want to delete, and then choose Remove Policy.

You can also use the Amazon SES API to delete sender authentication policies. The following procedure uses the AWS Command Line Interface (AWS CLI) to interact with the Amazon SES API. You can also interact with the API by using an AWS SDK, or by making HTTP requests directly.
Delete Data Related to Receiving Rules

If you use Amazon SES to receive incoming email, you can create receipt rules that are applied to one or more identities (email addresses or domains). These rules determine what Amazon SES does with incoming mail sent to the specified identities.

To delete a receipt rule by using the Amazon SES console
1. Open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. Under Email Receiving, choose Rule Sets.
3. If the receipt rule is part of the active rule set, choose View Active Rule Set. Otherwise, choose the rule set that contains the receipt rule that you want to delete.
4. In the list of receipt rules, choose the rule that you want to delete.
5. On the Actions menu, choose Delete.
6. On the confirmation dialog box, choose Delete.

You can also use the Amazon SES API to delete receipt rules. The following procedure uses the AWS Command Line Interface (AWS CLI) to interact with the Amazon SES API. You can also interact with the API by using an AWS SDK, or by making HTTP requests directly.

To delete a receipt rule by using the AWS CLI
• At the command line, type the following command:

```
aws ses delete-receipt-rule --rule-set myRuleSet --rule-name myReceiptRule
```  

In this command, replace `myRuleSet` with the name of the receipt rule set that contains the receipt rule. Replace `myReceiptRule` with the name of the receipt rule that you want to delete.

Delete Data Related to IP Address Filters

If you use Amazon SES to receive incoming email, you can create filters to explicitly accept or block messages that are sent from specific IP addresses.

To delete an IP address filter by using the Amazon SES console
1. Open the Amazon SES console at https://console.aws.amazon.com/ses/.
2. Under Email Receiving, choose IP Address Filters.
3. In the list of IP address filters, choose the filter that you want to remove, and then choose Delete.
You can also use the Amazon SES API to delete IP address filters. The following procedure uses the AWS Command Line Interface (AWS CLI) to interact with the Amazon SES API. You can also interact with the API by using an AWS SDK, or by making HTTP requests directly.

To delete an IP address filter by using the AWS CLI

- At the command line, type the following command:

  ```bash
  aws ses delete-receipt-filter --filter-name IPfilter
  ```

  In this command, replace `IPfilter` with the name of the IP address filter you want to delete.

Delete Data in Email Templates

If you use email templates for sending email, it's possible that those templates might contain personal data, depending on how you configured them. For example, you might have added an email address to the template that recipients could contact for more information.

You can only delete email templates by using the Amazon SES API.

To delete an email template by using the AWS CLI

- At the command line, type the following command:

  ```bash
  aws ses delete-template --template-name sampleTemplate
  ```

  In this command, replace `sampleTemplate` with the name of the email template that you want to delete.

Delete Data in Custom Verification Email Templates

If you use customized templates for verifying new email sending addresses, it's possible that those templates might contain personal data, depending on how you configured them. For example, you might have added an email address to the verification email template that recipients could contact for more information.

You can only delete custom verification email templates by using the Amazon SES API.

To delete a custom verification email template by using the AWS CLI

- At the command line, type the following command:

  ```bash
  aws ses delete-custom-verification-email-template --template-name verificationEmailTemplate
  ```

  In this command, replace `verificationEmailTemplate` with the name of the custom verification email template that you want to delete.
Delete All Personal Data by Closing Your AWS Account

It's also possible to delete all personal data that's stored in Amazon SES by closing your AWS account. However, this action also deletes all other data—personal or non-personal—that you have stored in every other AWS service.

When you close your AWS account, the data in your AWS account is retained for 90 days. After that retention period, it's deleted permanently and irreversibly.

**Warning**
Don't complete the following procedure unless you're certain that you want to completely remove all data that's stored in your AWS account across all AWS services and regions.

You can close your AWS account by using the AWS Management Console.

**To close your AWS account**

2. Go to the **Account Settings** page at https://console.aws.amazon.com/billing/home?#/account.

   **Warning**
The following two steps will permanently delete all of the data you've stored in all AWS services across all AWS Regions.

3. Under **Close Account**, read the disclaimer that describes the consequences of closing your AWS account. If you agree to the terms, select the check box, and then choose **Close Account**.
4. On the confirmation dialog box, choose **Close Account**.
Amazon SES frequently asked questions (FAQs)

This section contains answers to frequently asked questions related to using Amazon SES.

FAQs
- Amazon SES Sending review process FAQs (p. 489)
- DNS Blackhole List (DNSBL) FAQs (p. 503)
- Amazon SES email sending metrics FAQs (p. 506)

Amazon SES Sending review process FAQs

We monitor the email that's sent through Amazon SES to make sure that the service isn't being used to deliver malicious, unsolicited, or low-quality email. If we determine that a user is sending content that falls into one of these categories, we take actions on that account. We call this process our sending review process.

In many cases, when we detect an issue with an account, we place that account under review (p. 489). In other cases, we pause the account's ability to send email (p. 492). We take these actions to protect each account's sender reputation, and to prevent other Amazon SES users from experiencing service interruptions and deliverability issues.

This section contains frequently asked questions about the following topics:
- Account under review FAQ (p. 489)
- Sending pause FAQ (p. 492)
- Bounce FAQ (p. 494)
- Complaint FAQ (p. 496)
- Spamtrap FAQ (p. 500)
- Manual investigation FAQ (p. 502)

Account under review FAQ

Q1. I received a message stating that my account is under review. What does that mean?

We've detected an issue related to the email sent from your account, and we're giving you time to fix it. You can continue to send email as you normally would, but you should also correct the issue that caused your account to be placed under review. If you don't correct the issue before the review period is over, we might pause your ability to send additional email.

Q2. Will I always be notified if my account is placed under review?

Yes. You'll receive a notification at the email address associated with your AWS account.
Q3. Why didn't I receive a notification that my account is under review?

When your account is placed under review, we automatically send a notice to the email address associated with your AWS account. This email address is the one you specified when you created your AWS account. In some cases, this email address may be different from the one you use to send email using Amazon SES.

We recommend that you monitor your sender reputation by regularly consulting the Reputation Dashboard (p. 369). You can also set up automated alarms in Amazon CloudWatch (p. 382). These alarms can send you a notification when your reputation metrics exceed certain thresholds. You can also configure Amazon CloudWatch to contact you in other ways, such as by sending a text message to your mobile phone.

Q4. Will the fact that my Amazon SES account is under review impact my use of other AWS services?

You'll still be able to use other AWS services while your Amazon SES account is under review. However, if you request a service quota increase for another AWS service that sends outbound communications (such as Amazon SNS), that request may be denied until the review period for your Amazon SES account is lifted.

Q5. What should I do if my account is under review?

You should do the following:

- If your situation allows it, stop sending mail until you fix the problem. You can still send email while your account is under review. However, if you continue to send mail without making changes, you might inadvertently make the issue worse.
- Look at the email you received from us for a summary of the issue.
- Investigate your sending to determine what aspect of your sending specifically triggered the issue.
- After you make changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. In your message, provide detailed information about the steps you've taken to resolve the issue, and describe how these steps prevent the issue from happening again in the future.
- Be sure to provide any information we specifically request. We need this information to evaluate your case.

Q6. What's a review?

You can request that we review our decision to place your under review. See the following question for more information about requesting a review.

Q7. How do I request a review?

To request a review, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf.

In your request, provide the following information:

- Information about the root cause of the event that caused your account to be placed under review.
- A list of the changes that you've made to correct the issue. Only include the steps you've already implemented, not the steps you plan to implement in the future.
• Information about how these changes prevent the same issue from occurring again in the future.

Depending on the nature of the event that led us to place your account under review, we might require additional information. See the FAQ topic associated with the issue you experienced for a list of the information you should include in your request.

**Q8. What if my review request isn't accepted?**

We'll respond to your request with information about why we didn't accept it. In some cases, you'll be able to submit another request if you're able to demonstrate that you resolved the issue, and that your changes prevent the issue from occurring again in the future.

**Q9. Can you help me diagnose the problem?**

Typically we can give you only a high-level overview of your issue (for example, that you have a problem with bounces). You'll need to investigate the root cause on your end.

**Q10. How will I know if my account is no longer under review?**

The Reputation Dashboard includes information about the current status of your account. For more information, see [Using the reputation dashboard to track bounce and complaint rates](p. 369).

**Q11. Do you place my account under review every time there's a problem?**

No. In some situations, we might pause your account's ability to send email without first placing your account under review. For example:

• If the issue is very serious.
• If your account has been placed under review for the same issue multiple times in the past. For this reason, it's important to address the underlying problem rather than just resolve the specific incident that led to your account being placed under review. For instance, if a particular campaign caused us to place your account under review, you have to do more than simply stop that campaign. You should determine which properties of the campaign were problematic and ensure that you have processes in place so that your future campaigns don't have the same issue.

In either of these situations, we automatically send you a notification when we pause your account's ability to send email.

**Q12. What if I make my fixes shortly before the review period expires?**

Sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. In your reply to the case, let us know that you've resolved the issue.

**Q13. Can I get help from my AWS representative or Premium Support?**

If you're already working with an AWS account representative, we'll automatically contact him or her when your account is placed under review. Your account representative may be able to provide additional information to help you better understand the issue. If you use Premium Support, you should also contact that team for additional help.
Sending pause FAQ

Q1. I received a message stating that my account's ability to send email is paused. What does that mean?

We paused your account's ability to send email because of a critical issue with emails you sent. Most often, we pause accounts for one of the following reasons:

- We previously placed your account under review. The issues that caused us to place your account under review weren't corrected before the end of the review period, so we paused your account's ability to send email.
- We've placed your account under review several times for the same issue.
- Your account sent email that violated the AWS Service Terms. If these violations are serious, we might pause your account's ability to send email without placing your account under review first.

Q2. Will I always be notified if my account's ability to send email is paused?

Yes. You'll receive a notification at the email address associated with your AWS account.

Q3. My account's ability to send email is paused. Why didn't I receive a notification?

When we pause an account's ability to send email, we automatically send a notification to the email address associated with that account.

**Note**

When you create your AWS account, you must provide an email address. You can change this address at any time. For more information about changing the address associated with your AWS account, see Managing an AWS Account in the AWS Billing and Cost Management User Guide.

You can use Amazon CloudWatch to create alarms that inform you when your bounce and complaint rates are too high. Creating an alarm is a good way to receive an early warning of factors that could cause us to pause your account's ability to send email. However, there are factors other than bounces and complaints that could cause us to pause your ability to send email. For more information about creating alarms in CloudWatch, see Creating reputation monitoring alarms using CloudWatch (p. 382).

You can also use the Deliverability Dashboard (p. 369) to determine the current status of your account. For example, if your account's ability to send email is currently paused, the Account status section of the Deliverability Dashboard displays a status of SENDING PAUSE. If your account is able to send email normally, it displays a status of HEALTHY.

Finally, you can check the AWS Personal Health Dashboard (PHD) at https://phd.aws.amazon.com/ to determine if your account's ability to send email is currently paused. When we pause an account's ability to send email, we automatically add an SES sending paused event to the Event log section of the PHD. The SES sending paused event always has a Status of Closed, regardless of whether or not the account's ability to send email is currently paused. The event log also includes a copy of the email that we sent to the email address associated with your AWS account when the sending pause event occurred.

You can use CloudWatch to create an alarms that alert you when new events appear on your Personal Health Dashboard. For more information, see Monitoring AWS Health Events with CloudWatch Events in the AWS Health User Guide.
Q4. My account's ability to send email is paused. Does this impact my ability to use of other AWS services?

You can still use other AWS services while your account's ability to send email is paused. However, if you request a service quota increase for another AWS service that sends outbound communications (such as Amazon SNS), we might deny your request until your account's ability to send email is restored.

Q5. What should I do if my account's ability to send email is paused?

You should do the following:

- Look at the email you received from us for a summary of the issue.
- Investigate your sending to determine what aspect of your sending specifically triggered the issue.
- After you make changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. In your message, provide detailed information about the steps you've taken to resolve the issue, and describe how these steps prevent the issue from happening again in the future.
- Be sure to provide any information we specifically request. We need this information to evaluate your case.

Q6. What's a review?

You can request that we review our decision to place your under review. See the following question for more information about requesting a review.

Q7. How do I request a review?

To request a review, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf.

In your request, provide the following information:

- Information about what caused the issue.
- A list of the changes that you've made to correct the issue. Only include the steps that you've already implemented, not the steps you plan to implement in the future.
- Information about how these changes will prevent the same issue from occurring again in the future.

Depending on the nature of the event that led us to pause your account's ability to send email, we might require additional information. See the FAQ topic associated with the issue you experienced for a list of the information you should include in your request.

Q8. What if my request isn't accepted?

We'll respond to your request with information about why we didn't accept it. In some cases, you'll be able to submit another request if you're able to demonstrate that you resolved the issue, and that your changes prevent the issue from occurring again in the future.

Q9. Can you help me diagnose the problem?

Typically we can give you only a high-level overview of your issue (for example, that you have a problem with bounces). It's your responsibility to correct the issue.
Q10. How do I know if my account's ability to send email has been restored?

The Reputation Dashboard includes information about the current status of your account. For more information, see Using the reputation dashboard to track bounce and complaint rates (p. 369).

Q11. Can I get help from my AWS representative or Premium Support?

If you're already working with an AWS account representative, we'll automatically contact him or her if we pause your account's ability to send email. Your account representative may be able to provide additional information to help you better understand the issue. If you use Premium Support, you should also contact that team for additional help.

Bounce FAQ

Q1. Why do you care about my bounces?

High bounce rates are often used by entities such as email providers and anti-spam organizations to detect senders who engage in bad email-sending practices. High bounce rates can lead to email being sent to the spam folder rather than the inbox.

Q2. What should I do if I receive a notification stating that my account is under review or that my sending is paused because of my account's bounce rate?

Identify the cause of the issue, and then correct it. After you make changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. In your message, provide detailed information about the steps you've taken to resolve the issue, and describe how these steps prevent the issue from happening again in the future. Also include the following information:

- The method you use to track your bounces
- How you ensure that the email addresses of new recipients are valid prior to sending to them.
  For example, which of the recommendations are you following in Q11. What can I do to minimize bounces? (p. 496)

Q3. What types of bounces count toward my bounce rate?

Your bounce rate includes only hard bounces to domains you haven't verified. Hard bounces are permanent delivery failures such as "address does not exist." Temporary and intermittent failures such as "mailbox full," or bounces due to blocked IP addresses, don't count toward your bounce rate.

Q4. Do you disclose the bounce rates that could cause my account to be placed under review or that could cause my sending to be paused?

For best results, you should maintain a bounce rate below 2%. Higher bounce rates can impact the delivery of your emails.
If your bounce rate is 5% or greater, we'll place your account under review. If your bounce rate is 10% or greater, we might pause your account's ability to send additional email until you resolve the issue that resulted in the high bounce rate.

Q5. Over what period of time is my bounce rate calculated?

We don't calculate your bounce rate based on a fixed period of time, because different senders send at different rates. Instead, we look at a representative volume—an amount of email that represents your typical sending practices. To be fair to both high- and low-volume senders, the representative volume is different for each user and changes as the user's sending patterns change.

Q6. Can I calculate my own bounce rate by using the information from the Amazon SES console or the GetSendStatistics API?

No. The bounce rate is calculated using representative volume (see Q5. Over what period of time is my bounce rate calculated? (p. 495)). Depending on your sending rate, your bounce rate can stretch farther back in time than the Amazon SES console or GetSendStatistics can retrieve. In addition, only emails to non-verified domains are considered when calculating your bounce rate. However, if you regularly monitor your bounce rates using those methods, you should still have a good indicator that you can use to catch problems before they get to levels that cause us to place your account under review or pause your account's ability to send email.

Q7. How can I find out which email addresses bounced?

Examine the bounce notifications that Amazon SES sends you. The email address to which Amazon SES forwards the notifications depends on how you sent the original messages, as described at Amazon SES notifications sent by email (p. 265). You can also set up bounce notifications through Amazon Simple Notification Service (Amazon SNS), as described at Monitoring Amazon SES email sending using notifications (p. 265). Note that simply removing bounced addresses from your list without any additional investigation might not solve the underlying problem. For information about what you can do to reduce bounces, see Q11. What can I do to minimize bounces? (p. 496).

Q8. If I haven't been monitoring my bounces, can you give me a list of addresses that have bounced?

No, we can't provide a complete list of addresses that have bounced. You are responsible for monitoring and acting upon the bounces for your account.

Q9. How should I handle bounces?

You need to remove bounced addresses from your mailing list and stop sending mail to them immediately. If you're a small sender, it might be sufficient to simply monitor bounces through email and manually remove bounced addresses from your mailing list. If your volume is higher, you'll probably want to set up automation for this process, either by programmatically processing the mailbox where you receive bounces, or by setting up bounce notifications through Amazon SNS. For more information, see Monitoring Amazon SES email sending using notifications (p. 265).

Q10. Could my emails be bouncing because I've reached my sending quota?

No. Bounces aren't related to sending quotas. If you try to exceed your sending quota, you'll receive an error from the Amazon SES API or SMTP interface when you try to send an email.
Q11. What can I do to minimize bounces?

First, be sure that you're aware of your bounces (see Q7. How can I find out which email addresses bounced? (p. 495)). Then follow these guidelines:

- Don’t buy, rent, or share email addresses. Send email only to recipients who explicitly requested to receive email from you.
- Remove bounced email addresses from your list.
- On web forms, ask users to enter their email addresses two times, and check to make sure both addresses match before the form can be submitted.
- Use double opt-in to sign up new users. That is, when a new user signs up, send them a confirmation email that they need to click before receiving any additional mail. This prevents people from signing up other people as well as accidental sign-ups.
- If you must send to addresses that you haven’t mailed lately (and thus you can't be confident that the addresses are still valid), do so only with a small portion of your overall sending. For more information, see our blog post Never send to old addresses, but what if you have to?
- Ensure that you're not structuring sign-ups to encourage people to use fictional addresses. For example, don't provide any added value or benefits until recipients verify their addresses.
- If you have an "email a friend" feature, use CAPTCHA or a similar mechanism to discourage automated use of the feature, and don't allow users to insert arbitrary content.
- If you’re using Amazon SES for system notifications, ensure that you're sending the notifications to real addresses that can receive mail. Also consider turning off notifications that you don't need.
- If you're testing a new system, be sure you're either sending to real addresses that can receive email, or you're using the Amazon SES mailbox simulator. For more information, see Testing email sending in Amazon SES (p. 181).

Complaint FAQ

Q1. What's a complaint?

A complaint occurs when a recipient reports that they don't want to receive an email. They might have clicked the "Report spam" button in their email client, complained to their email provider, notified Amazon SES directly, or through some other method. This topic includes general information about complaints. If your notification contains specific information about the source of the complaints, also read the relevant topic: Amazon SES complaints through feedback loops FAQ (p. 497), Amazon SES complaints directly from recipients FAQ (p. 499), or Amazon SES complaints through email providers FAQ (p. 500).

Q2. Why do you care about my complaints?

High complaint rates are often used by entities such as email providers and anti-spam organizations as indicators that a sender is sending to recipients who didn't specifically sign up to receive emails, or that the sender is sending content that is different from the type that recipients signed up for.

Q3. What should I do if I receive a notice saying that my account is under review or that my sending is paused because of an issue with complaints?

Review your list acquisition process and the content of your emails to try to understand why your recipients might not appreciate the email they're receiving from you. Identify the cause of the issue, and then correct it. After you make changes that you believe will resolve the issue, sign into the AWS Console.
and go to Support Center. Reply to the case we opened on your behalf. In your message, provide detailed information about the steps you’ve taken to resolve the issue, and describe how these steps prevent the issue from happening again in the future.

Q4. What can I do to minimize complaints?

First, be sure that you monitor the complaints that Amazon SES can notify you about, which are complaints that Amazon SES receives through feedback loops (see the Amazon SES complaints through feedback loops FAQ (p. 497)). Then follow these guidelines:

- Do not buy, rent, or share email addresses. Use only addresses that specifically requested your mail.
- Use double opt-in to sign up new users. That is, when users sign up, send them a confirmation email that they need to click before receiving any additional mail. This prevents people from signing up other people as well as accidental sign-ups.
- Monitor engagement with the mail you send and stop sending to recipients who don't open or click your messages.
- When new users sign up, be clear about the type of email they will receive from you, and ensure that you send only the type of mail that they signed up for. For example, if users sign up for news updates, don't send them advertisements.
- Ensure that your mail is well-formatted and looks professional.
- Ensure that your mail is clearly from you and can't be confused for something else.
- Provide users an obvious and easy way to unsubscribe from your mail.

Amazon SES complaints through feedback loops FAQ

This topic provides information about complaints that Amazon SES receives from email providers through feedback loops. For general information that applies to all types of complaints, see the Complaint FAQ (p. 496).

Q1. How is this type of complaint reported?

Most email client programs provide a button labeled "Mark as Spam" or similar, which moves the message to a spam folder and forwards it to the email provider. Additionally, most email providers maintain an abuse address (such as abuse@example.com), where users can forward unwanted email and request that the provider take action to prevent them. If the Amazon SES has a feedback loop (FBL) set up with the email provider, then they send the complaint back to Amazon SES.

Q2. Are these complaints included in the complaint rate statistic shown in the Amazon SES console and returned by the GetSendStatistics API?

Yes. However, the complaint rate statistic doesn't include complaints from email providers that don't provide feedback to Amazon SES. The complaint rate from domains that provide feedback is likely to be representative of the rest of your sending as well.

Q3. How can I be notified of these complaints?

You can be notified through email or through Amazon SNS notifications. See the set-up instructions in Monitoring Amazon SES email sending using notifications (p. 265).

Q4. What should I do if I receive a complaint notification through email or through Amazon SNS?

First, you need to remove addresses that generated complaints from your mailing list and stop sending mail to them immediately. Do not even send an email that says you've received the request to unsubscribe. Consider setting up automation for this process, either by programmatically processing the
mailbox where you receive complaints, or by setting up complaint notifications through Amazon SNS. For more information, see Monitoring Amazon SES email sending using notifications (p. 265).

Then, take a close look at your sending to determine why your recipients don't appreciate the mail you're sending, and address that underlying problem. For every person who complains, there are potentially dozens who didn't appreciate your mail who didn't (or weren't able to) complain. If you only remove the recipients who actually complain, you're not addressing the underlying problem.

**Q5. Do you disclose the Amazon SES complaint rates that could cause my account to be placed under review or that could result in my account's ability to send email being paused?**

For best results, you should maintain a complaint rate below 0.1%. Higher complaint rates can impact the delivery of your emails.

If your complaint rate is 0.1% or greater, we'll place your account under review. If your complaint rate is 0.5% or greater, we might pause your account's ability to send additional email until you resolve the issue that resulted in the high complaint rate.

**Q6. Over what period of time is my complaint rate calculated?**

We don't calculate your complaint rate based on a fixed period of time, because different senders send at different rates. Instead, we look at a representative volume—an amount of mail that represents your typical sending practices. To be fair to both high- and low-volume senders, the representative volume is different for each user and changes as the user's sending patterns change. Additionally, the complaint rate isn't calculated based on every email. Instead, it's calculated as the percentage of complaints on mail sent to domains that send complaint feedback to Amazon SES.

**Q7. Can I calculate my own complaint rate by using metrics from the Amazon SES console or the GetSendStatistics API?**

No. There are two primary reasons for this:

- The complaint rate is calculated using representative volume (see Q6. Over what period of time is my complaint rate calculated? (p. 498)). Depending on your sending rate, your complaint rate can stretch farther back in time than the Amazon SES console or GetSendStatistics API can retrieve. For this reason, we recommend that you regularly use these methods to monitor the complaint rate for your account. Monitoring your complaint rate in this way gives you the information you need to identify problems before they reach levels that could impact the delivery of your email.
- When calculating complaint rate, not every email counts. Complaint rate is calculated as the percentage of complaints on mail sent to domains that send complaint feedback to Amazon SES.

**Q8. How can I find out which email addresses complained?**

Examine the complaint notifications that Amazon SES sends you through email or through Amazon SNS (see Monitoring Amazon SES email sending using notifications (p. 265)). However, different email providers provide differing amounts of information, and some providers redact the recipient's email address before passing the complaint notification to Amazon SES. To enable you to find the recipient's email address in the future, your best option is to store your own mapping between an identifier and the Amazon SES message ID that Amazon SES passes back to you when it accepts the email. Note that Amazon SES doesn't retain any custom message IDs that you add.

**Q9. If I haven't been monitoring my complaints, can you give me a list of addresses that have complained?**

Unfortunately, we can't give you a comprehensive list. However, you can monitor future complaints by email or through Amazon SNS.
Q10. Can I get a sample email?

We can't send you a sample email upon request, but you might find this information in the complaint notification. For more information, see Q8. How can I find out which email addresses complained? (p. 498).

Amazon SES complaints directly from recipients FAQ

This topic provides information about complaints that Amazon SES receives directly from recipients. For general information that applies to all types of complaints, see the Complaint FAQ (p. 496).

Q1. How is this type of complaint reported?

Multiple recipients directly contacted Amazon SES about your mail through email or some other means.

Q2. Are these complaints included in the complaint rate statistic shown in the Amazon SES console and returned by the GetSendStatistics API?

No. The complaint rate statistic you retrieve using the Amazon SES console or the GetSendStatistics API only includes complaints that Amazon SES receives through feedback loops. For more information about those types of complaints, see the Amazon SES complaints through feedback loops FAQ (p. 497).

Q3. Why haven't I heard about these complaints through email feedback notifications or through Amazon SNS?

Email feedback forwarding and Amazon SNS notifications only include complaints that Amazon SES receives through feedback loops. You won't receive notifications for complaints that recipients filed directly with Amazon SES.

Q4. How can I find out which email addresses complained?

To protect the identities of the recipients who complained, we can't list the email addresses that complained about your email.

Rather than focus on removing individual recipients from your lists, we recommend that you determine the problem that led to the complaints being issued. We recommend that you begin by reviewing your customer acquisition process, and that you remove any customers from your lists that didn't explicitly ask to receive email from you. You should also analyze the content of your emails to try to understand why your recipients are complaining.

Q5. Can I get a sample email?

To protect the identities of the recipients who complained, we can't provide copies of the emails that caused your recipients to complain.

Q6. What should I do if I receive a notification stating that my account is under review or that my sending is paused because of direct complaints?

Immediately change your sending processes so that you're only sending messages recipients who have specifically signed up to receive them. Also, ensure that you're sending the type of content that your recipients signed up to receive. After you make changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. In your message, provide detailed information about the steps you've taken to resolve the issue, and describe how these steps prevent the issue from happening again in the future.

If you don't request a review within three weeks, and we continue to receive direct recipient complaints, we might pause your account's ability to send email.
Amazon SES complaints through email providers FAQ

This topic provides information about complaints that Amazon SES receives through email providers (also called mailbox providers). For general information that applies to all types of complaints, see the Complaint FAQ (p. 496).

Q1. How is this type of complaint reported?

An email provider reported to Amazon SES that a significant number of its customers marked your emails as spam. The report was provided to Amazon SES through a means other than the feedback loops described in the Amazon SES complaints through feedback loops FAQ (p. 497).

Q2. Are these complaints included in the complaint rate statistic shown in the Amazon SES console and returned by the GetSendStatistics API?

No. The complaint rate statistic you retrieve using the Amazon SES console or the GetSendStatistics API only includes complaints that Amazon SES receives through feedback loops.

Q3. Why haven't I heard about these complaints through email feedback notifications or through Amazon SNS?

Email feedback forwarding and Amazon SNS notifications only include complaints that Amazon SES receives through feedback loops.

Q4. How can I find out which email addresses complained?

Email providers typically don't disclose this information. However, rather than focusing on removing individual recipients from your list, you need to focus on finding and fixing the underlying problem. Start by reviewing your list acquisition process and the content of your emails to try to understand why your recipients might not appreciate your email.

Q5. Can I get a sample email?

No. Email providers typically don't provide an example email.

Q6. What should I do if I receive a notification stating that my account is under review or that my sending is paused because of email provider complaints?

Identify the cause of the issue, and then correct it. After you make changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. In your message, provide detailed information about the steps you've taken to resolve the issue, and describe how these steps prevent the issue from happening again in the future. If you don't request a review within three weeks, and we continue to receive complaints from providers, we might pause your account's ability to send additional email.

Spamtrap FAQ

Q1. What are spamtraps?

A spamtrap is a special email address maintained by an Internet Service Provider (ISP), email provider, or anti-spam organization. Because that address will never legitimately be signed up to receive email, the organizations that maintain these spamtraps know that anyone who sends mail to any of these addresses is likely to be engaging in questionable email practices.
Q2. How are spamtraps set up?

Spamtrap addresses can be set up in multiple ways. They can be converted from addresses that were once valid, but have been unused (and bouncing) for an extended period of time. They can also be addresses that were set up just to be spamtraps. They can be unusual addresses that are hard to guess, and sometimes they are addresses that are close to real addresses (for example, introducing a typo into a common domain name). Often, but not always, spamtraps are "seeded" into the world by putting them on the internet in a variety of ways.

Q3. How does Amazon SES know if I am sending to spamtraps?

Certain organizations that operate spamtraps send Amazon SES notifications when their spamtraps are hit by Amazon SES senders.

Q4. How does Amazon SES use the spamtrap reports?

We review the reports. If we determine that your account is sending email to spamtraps, we place your account under review and ask you to fix the underlying problem. If you don't fix the problem before the review period is over, we might pause your account's ability to send additional email. If your spamtrap problem is very severe, we might pause your account's ability to send email immediately, without placing your account under review first.

Q5. What should I do if a receive a notice saying that my account is under review or that my sending is paused because of an issue with spamtraps?

First, you should address the issue that caused us to place your account under review or pause your ability to send email. Next, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. In your message, provide detailed information about the steps you've taken to resolve the issue, and describe how these steps prevent the issue from happening again in the future. If we agree that the changes you've made appropriately address the issue, we'll cancel the review period or remove the sending pause from your account.

Because of the way that spamtrap hits are reported, it may take three weeks or more before we are able to determine if the changes you made solved the issue.

Q6. How many spamtrap hits can I have before you place my account under review or pause my account's ability to send email?

We don't disclose the specific number of spamtrap hits that cause us to take action on your account. However, it's important to note that even a small number of spamtrap hits can have a very negative effect on your reputation as a sender, so you should take spamtrap reports seriously.

Q7. Do you disclose the spamtrap addresses?

No. In order for spamtraps to be effective, it's essential that they remain confidential. Spamtrap organizations disclose only the occurrence of spamtrap hits, not the actual spamtrap addresses.

Q8. What can I do to avoid sending to spamtraps?

To reduce the risk of sending to spamtraps, follow these guidelines:
• Do not buy, rent, or share email addresses. Use only addresses that specifically requested your mail.
• On web forms, ask users to enter their email addresses two times, and check to make sure both addresses match before the form can be submitted.
• Use double opt-in to sign up new users. That is, when users sign up, send them a confirmation email that they need to click before receiving any additional mail.
• Ensure that you remove addresses that hard bounce from your list, so that they are removed long before they are converted to spamtraps.
• Ensure that you're monitoring engagement by your recipients, and stop sending to recipients who haven't engaged with your emails or website recently. Time frames for what an "engaged user" is depend on your use case, but generally speaking if users haven't opened or clicked your emails in several months, you should consider removing them unless you have evidence that they do want your mail.
• Be very careful with re-engagement campaigns where you intentionally contact people who haven't interacted with you recently. These efforts tend to be highly risky, and can often cause problems not only with spamtrap sending, but also with bounces and complaints.
• Send an opt-in message to your entire mailing list and keep only the recipients who click on the verification link. In addition to removing inactive recipients from your list, this procedure also helps remove spamtrap addresses. However, we don't recommend using this technique if you think that your mailing list might contain a lot of bad addresses, or if your account already has a problem with bounces, because it might cause your account's bounce rate to increase further.

Manual investigation FAQ

Q1. What should I do if I receive a notification stating that my account is under review or that my sending is paused because of a manual investigation?

An Amazon SES investigator has identified a significant problem with your sending. Typical problems include, but aren't limited to, the following:

• Your sending violates the AWS Acceptable Use Policy (AUP).
• Your emails appear to be unsolicited.
• Your content is phishing related (this includes simulated phishing).
• Your content is otherwise associated with a use case that Amazon SES doesn't support.

If we believe that the problem can be corrected, we place your account under review for a certain amount of time. While your account is under review, you should make changes to your email sending practices to correct the issue.

If we don't believe that the problem can be corrected, or if the problem is very severe, we might pause your account's ability to send email without first placing your account under review.

Q2. What issues could cause you to perform a manual review of my email sending?

There are several issues that could cause us to begin a manual review of your account. These reasons include, but aren't limited to, the following:

• Recipients contact Amazon SES to complain about email sent from your account.
• We detect unusual changes in your email sending patterns.
• Our spam filters find characteristics of your email that are typical of unsolicited or low-quality content.

When we place your account under review or pause your account's ability to send email, we send you a notification. In most cases, this notification contains information about the issue, and provides information about the next steps you can take.

Q3. What are "unsolicited" emails?

Unsolicited emails are emails that the recipient didn't explicitly ask to receive. This includes cases in which a recipient signs up for a certain type of mail (for example, notifications), and instead is sent a different type of mail (for example, advertisements).

When we place your account under review or pause your account's ability to send email, we send you a notification. If you receive a notification stating that we're taking one of these actions because of an issue with unsolicited email, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. In your message, include the following information:

• Are all the messages that you send specifically requested by the recipient, and do they comply with the AWS Acceptable Use Policy?
• Have you acquired email addresses in any way other than a customer specifically interacting with you or your website and requesting emails from it? You should explain how you acquired your mailing list.
• How do your subscribe and unsubscribe processes work? You should include your opt-in and opt-out links.

Q4. What should I do if I receive a notification stating that my account is under review or that my sending is paused because of a manual review?

Identify the cause of the issue, and then correct it. After you make changes that you believe will resolve the issue, sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf. In your message, provide detailed information about the steps you've taken to resolve the issue, and describe how these steps prevent the issue from happening again in the future. If we agree that the changes you've made appropriately address the issue, we'll cancel the review period on your account.

Q5. What types of problems do you view as "correctable?"

Generally, we believe the situation is correctable if you have a history of good sending practices, and if there are steps you can take to eliminate the problematic sending while continuing the bulk of your sending. For example, if you're sending three different types of email and only one type is problematic, you might be able to simply stop the problematic sending and continue with the rest of your sending.

Q6. What if I can't find the source of the problem?

You can sign into the AWS Console and go to Support Center. Reply to the case we opened on your behalf and request a sample of the mail that caused the issue.

DNS Blackhole List (DNSBL) FAQs

Domain Name System-based Blackhole Lists (DNSBLs)—sometimes referred to as Realtime Blackhole Lists (RBLs), deny lists, blocklists, or blacklists—are intended to inform email providers of IP addresses that are suspected of sending unwanted email.
Different DNSBLs have different impacts on email deliverability. This topic describes how DNSBLs impact the delivery of emails you send using Amazon SES, as well as our policies for removing Amazon SES IP addresses from DNSBLs.

**Note**
This topic is about the DNSBLs that email providers use to block incoming messages. For information about how Amazon SES blocks outgoing email sent to recipients whose email addresses have previously generated bounces, see Using the Amazon SES global suppression list (p. 192).

**Q1. How do DNSBLs impact email delivery?**

Different DNSBLs have different impacts on the successful delivery of a message. Major email providers—including Gmail, Hotmail, AOL, and Yahoo—seem to recognize a very small number of highly regarded DNSBLs, such as those offered by Spamhaus. In our experience, other DNSBLs tend to have a low impact, although some mail systems emphasize certain DNSBLs over others.

Finally, many email providers have their own internal deny lists. Email providers guard these lists very closely, and rarely share them with the public. If an IP address is on one of these lists, it can have a major impact on your ability to send email to recipients who use that provider.

**Q2. How do IP addresses end up on DNSBLs?**

There are several ways that an IP address can end up on a DNSBL. IP addresses can be added to DNSBLs when they send email to a *spamtrap*. A spamtrap is an email address that doesn't belong to a human user. Spamtraps exist solely to collect spam and identify spammers. Some DNSBLs also allow individual users to submit IP addresses. A few DNSBLs even allow users to submit entire IP address ranges. Other DNSBLs are maintained through contributions by email administrators, and can include IP addresses that administrators believe are abusing their own systems.

**Q3. How does Amazon SES prevent its IP addresses from appearing on DNSBLs?**

Our systems look for signs of abuse. If we detect sending patterns or other characteristics that could lead to an IP address being added to a DNSBL, we send a notification to the sender. If the situation is severe, or if the sender doesn't fix the issue after we send the notification, we'll pause the sender's ability to send email until they resolve the issue. Enforcing our sending policies in this way helps reduce the chances that our IP addresses end up on DNSBLs.

**Q4. Can Amazon SES have its IP addresses removed from a DNSBL?**

We actively monitor DNSBLs that could impact delivery across the entire Amazon SES service, or that could impact the ability to send email to recipients who use major email providers, such as Gmail, Yahoo, AOL, and Hotmail. The DNSBLs offered by Spamhaus fall into this category. When one of our IP addresses appears on a list that meets either of these criteria, we take immediate action to have that address removed from the DNSBL as quickly as possible.

We don't monitor DNSBLs that are unlikely to impact delivery across the entire Amazon SES service, or that don't have a measurable impact on delivery to major email providers. The DNSBLs offered by SORBS and UCEPROTECT fall into this category. Because of the specific listing and delisting practices of the vendors who operate these lists, we are unable to have our IP addresses removed from these lists.
Q5. An email provider is rejecting my email because the sending IP address is listed by a DNSBL other than Spamhaus. What can I do?

First, confirm that the message was truly blocked because of a DNSBL. If your email was rejected because the sending IP address was added to a DNSBL, you'll receive a bounce notification that mentions the DNSBL provider by name, as in the following example:

```
554 5.7.1 Service unavailable; Client host [192.0.2.0] blocked using DNSBLName;
See: http://www.example.com/query/ip/192.0.2.0
```

If you received a bounce notification, but it didn't contain information similar to the message shown in the preceding example, then the email provider most likely rejected your message for a reason unrelated to being added to a DNSBL.

If you can confirm that an email provider is blocking your email because the sending IP address is on a DNSBL, there are a few things you can do:

- Contact the postmaster of the domain that rejected your message to request an exception from their spam filtering policy. Some postmasters have support processes, and may publish a postmaster page that describes this process. If the domain you're trying to contact doesn't publish its postmaster support policies, you might be able to contact the postmaster by sending email to postmaster@example.com, where example.com is the domain in question. Domains are required by RFC 5321 to have a postmaster mailbox.

When you contact the postmaster, provide the bounce codes you received, the headers of the email you're trying to send, a measurement of the impact the DNSBL is having on the delivery of your email, and information about why you believe that your email is being improperly blocked. The more information you can provide to the postmaster to demonstrate that you're sending legitimate email, the more likely the postmaster is to make an exception for you.

- If the email provider doesn't respond, or is unwilling to change their policies, consider using a dedicated IP address (p. 173). Dedicated IP addresses are addresses that only you can use. By implementing good sending practices, you can keep your engagement rates high, and your rates of bounces, complaints, and spamtrap hits low. Good sending practices can help ensure that your addresses don't end up on DNSBLs.

Q6. Email that I send to Gmail, Yahoo, Hotmail, or another major provider is being sent to the spam folder. Is this happening because my sending IP address is on a DNSBL?

Probably not. If an IP address is listed by a DNSBL with significant impact, such as one of the DNSBLs from Spamhaus, major email providers will reject email from that IP address completely, rather than sending it to the spam folder.

When major email providers accept an email (rather than rejecting it), they usually consider user engagement when considering whether to place the message in the inbox or in the spam folder. User engagement refers to the ways in which users interacted with the messages you sent them previously.

To increase the chances that your messages reach your customers' inboxes, you should implement all of the following best practices:
• **Never** rent or purchase lists of email addresses. Renting or purchasing lists is a violation of the AWS Acceptable Use Policy (AUP) and isn't allowed on Amazon SES under any circumstances.

• Only send email to customers who explicitly asked to receive email from you. In many countries and jurisdictions around the world, it's illegal to send email to recipients who didn't explicitly agree to receive email from you.

• Stop sending email to customers who haven't opened or clicked links in messages that you've sent in the past 30–90 days. This step can help to keep your engagement rates high, which increases the chances that the messages you send in the future arrive in recipients' inboxes.

• Use consistent design elements and writing styles in each message that you send to ensure that customers can easily identify messages from you.

• Use email authentication mechanisms, such as SPF (p. 129) and DKIM (p. 130).

• When customers use a web form to subscribe to your content, send them an email to confirm that they want to receive email from you. Don't send them any additional email until they confirm that they want to receive email from you. This process is known as **confirmed opt-in** or **double opt-in**.

• Make it easy for your customers to unsubscribe, and honor unsubscribe requests immediately.

• If you send email that contains links, check those links against the Spamhaus Domain Block List (DBL). To test your links, use the Domain Lookup Tool on the Spamhaus website.

By implementing these practices, you can improve your sender reputation, which increases the likelihood that the email you send reaches recipients' inboxes. Implementing these practices also helps keep the bounce and complaint rates low for your account, and reduces the risk of sending email to spamtraps.

---

### Amazon SES email sending metrics FAQs

Amazon SES collects several metrics about the emails you send. These metrics enable you to analyze the effectiveness of your email program and monitor important statistics, such as your bounce and complaint rates.

This section contains FAQs on the following topics related to email sending metrics:

- General Questions (p. 506)
- Open Tracking (p. 507)
- Click Tracking (p. 508)

### General Questions

**Q1. After an email is delivered, how long does Amazon SES continue to collect open and click metrics?**

Amazon SES collects open and click metrics for 60 days after each email is sent.

**Q2. If a user opens an email multiple times, or clicks a link in an email multiple times, is each of those events tracked separately?**

If a recipient opens an email multiple times, Amazon SES counts each open as a unique open event. Similarly, if a recipient clicks the same link multiple times, Amazon SES counts each click as a unique click event.
**Q3. Are open and click metrics aggregated, or can they be measured down to the recipient level?**

Opens and clicks are tracked at the recipient level. With open and click tracking, you can determine which recipients opened an email or clicked a link in an email.

**Q4. Can I retrieve open and click metrics using the Amazon SES API?**

The Amazon SES API does not provide a method for retrieving open and click metrics. However, you can retrieve open and click metrics for Amazon SES using the CloudWatch API. For example, you can use the AWS CLI to retrieve click metrics using the CloudWatch API by issuing the following command:

```bash
aws cloudwatch get-metric-statistics --namespace AWS/SES --metric-name Click --statistics Sum --period 86400 --start-time 2017-01-01T00:00:00Z --end-time 2017-12-31T23:59:59Z
```

The command shown above retrieves the total number of click events for each day in 2017. To retrieve open metrics change the value of the `metric-name` parameter to `Open`. You can also modify the `start-time` and `end-time` parameters to change the analysis period, or change the `period` parameter for more fine-grained analysis.

---

**Open Tracking**

**Q1. How does open tracking work?**

At the bottom of each email sent through Amazon SES, we insert a 1 pixel by 1 pixel transparent GIF image. Each email includes a unique reference to this image file; when the image is opened, Amazon SES can tell exactly which message was opened and by whom.

The addition of this tracking pixel does not change the appearance of your email.

**Q2. Is open tracking enabled by default?**

Open tracking is available to all Amazon SES users by default. To use open tracking, you must do the following:

1. Create a configuration set.
2. In the configuration set, create an event destination.
3. Configure the event destination to publish open event notifications to a destination.
4. In every email for which you want to track opens, specify the configuration set that you created in step 1.

For a more detailed explanation of this process, see the section called “Monitoring sending using event publishing” (p. 288).

**Q3. Can I omit the open tracking pixel from certain emails?**

There are two ways to omit the open tracking pixel from your emails. The first method is to send the email without specifying a configuration set. Alternatively, you can specify a configuration set that is not configured to publish data about open events.
Q4. Do you track opens for plaintext emails?

Open tracking only works with HTML emails. Because open tracking relies on the inclusion of an image, it is not possible to collect open metrics for users who open emails using a text-only (non-HTML) email client.

Click Tracking

Q1. How does click tracking work?

To track clicks, Amazon SES modifies each link in the body of the email. When recipients click a link, they are sent to an Amazon SES server, and are immediately forwarded to the destination address. As with open tracking, each redirect link is unique. This enables Amazon SES to determine which recipient clicked the link, when they clicked it, and the email from which they arrived at the link.

Important

If you send a single message to multiple recipients, each recipient will save the same click tracking link. To track individual recipients' click activity, send email to one recipient per send operation.

Q2. Can I disable click tracking?

You can disable click tracking for individual links by adding an attribute, \texttt{ses:no-track}, to the anchor tags in the HTML body of your email. For example, if you link to the AWS home page, a normal anchor link resembles the following:

\begin{verbatim}
<a href="https://aws.amazon.com">Amazon Web Services</a>
\end{verbatim}

To disable click tracking for that link, modify it to resemble the following:

\begin{verbatim}
<a \texttt{ses:no-track} href="aws.amazon.com">Amazon Web Services</a>
\end{verbatim}

Because \texttt{ses:no-track} isn't a standard HTML attribute, Amazon SES automatically removes it from the version of the email that arrives in your recipients' inboxes.

You can also disable click tracking for all messages that you send using a specific configuration set. To disable click tracking, modify the configuration set event destination so that it doesn't capture click events. For more information, see Managing event destinations in Amazon SES (p. 254).

Q3. How many links can be tracked in each email?

The click tracking system can track a maximum of 250 links.

Q4. Are click metrics collected for links in plain text emails?

It's only possible to track clicks in HTML emails.

Q5. Can I tag links with unique identifiers?

You can add an unlimited number of tags, as key-value pairs, to links in your email by using the \texttt{ses:tags} attribute. When you use this attribute, specify the keys and values using the same format that you would use to pass inline CSS properties: type the key, followed by a colon (\texttt{:)}, followed by the value. If you need to pass several key-value pairs, separate each pair with a semicolon (\texttt{;}).
For example, assume you want to add the tags `product:book`, `genre:fiction`, `subgenre:scifi`, `type:newrelease` to a link. The resulting link resembles the following:

```
<a ses:tags="product:book;genre:fiction;subgenre:scifi;type:newrelease;"
href="http://www.amazon.com/…/">New Releases in Science Fiction</a>
```

These tags are passed through to your event publishing destination so that you can perform additional analysis on the specific links that your users clicked.

**Note**
Link tags can include the numbers 0–9, the letters A–Z (both uppercase and lowercase), hyphens (-), and underscores (_).

**Q6. Do tracked links use the HTTP or HTTPS protocol?**

Tracking links use the same protocol as the original links in your email.

For example, if your email includes a link to `https://www.amazon.com`, the link is replaced with a tracking link that uses the HTTPS protocol. If your email includes a link to `http://www.example.com`, the link is replaced with a tracking link that uses HTTP. If your email includes both of the previously mentioned links, the HTTPS link is replaced with a tracking link that uses the HTTPS protocol, and the HTTP link is replaced with a tracking link that uses the HTTP protocol.

**Q7. A link in my email isn't being tracked. Why not?**

Amazon SES expects the links in your emails to contain properly encoded URLs. Specifically, URLs in your links must comply with [RFC 3986](https://tools.ietf.org/html/rfc3986). If a link in an email isn't properly encoded, recipients will still see the link in the email, but Amazon SES won't track click events for that link.

Issues related to improper encoding typically occur in URLs that contain query strings. For example, if the URL of a link in your email contains a non-encoded space character in the query string (such as the space between "John" and "Doe" in the following example: `http://www.example.com/path/to/page?name=John Doe`), Amazon SES won't track that link. However, if the URL uses an encoded space character instead (such as "%20" in the following example: `http://www.example.com/path/to/page?name=John%20Doe`), Amazon SES tracks it as expected.
Amazon SES resources

The following table lists resources that you might find useful as you work with Amazon Simple Email Service (Amazon SES).

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Simple Email Service API Reference</td>
<td>The Amazon SES API Reference. Contains complete descriptions of the API actions, parameters, and data types, and a list of errors that the service returns.</td>
</tr>
<tr>
<td>Amazon SES Pricing</td>
<td>Pricing information for Amazon SES.</td>
</tr>
<tr>
<td>SES Sending Limits Increase case</td>
<td>The Support Center form to request an increase to your sending quotas and move out of the sandbox.</td>
</tr>
<tr>
<td>Amazon SES Forum</td>
<td>A forum where Amazon SES users can post questions and discuss various Amazon SES topics.</td>
</tr>
<tr>
<td>AWS Messaging and Targeting Blog</td>
<td>The blog that contains blog posts and announcements by the Amazon SES team.</td>
</tr>
<tr>
<td>AWS Developer Tools</td>
<td>Links to developer tools and resources that provide documentation, code samples, release notes, and other information to help you build innovative applications with AWS.</td>
</tr>
<tr>
<td>AWS Support Center</td>
<td>The hub for creating and managing your AWS Support cases. Also includes links to other helpful resources, such as forums, technical FAQs, service health status, and AWS Trusted Advisor.</td>
</tr>
<tr>
<td>Contact Us</td>
<td>A central contact point for inquiries concerning AWS billing, account, events, abuse, and other issues.</td>
</tr>
<tr>
<td>AWS Glossary</td>
<td>The AWS Glossary. Contains definitions of common terms used in Amazon SES and other AWS services.</td>
</tr>
<tr>
<td>Conditions of Use</td>
<td>AWS Acceptable Use Policy. Describes email abuse and other prohibited uses of the web services offered by Amazon Web Services, Inc.</td>
</tr>
<tr>
<td><a href="mailto:email-abuse@amazon.com">email-abuse@amazon.com</a></td>
<td>An email address for reporting malicious or unsolicited (spam) email sent from Amazon SES.</td>
</tr>
</tbody>
</table>

When you contact this address, please provide the following information:

- The full headers of the email message. For more information about retrieving email headers, see https://support.google.com/mail/answer/22454?hl=en.
- The type of abuse you are experiencing. For example, unsolicited emails that do not provide a method of opting out.
This appendix contains supplementary information about sending emails through Amazon Simple Email Service (Amazon SES).

- For the header field requirements for emails that you send through Amazon SES, see Header fields (p. 511).
- For a list of attachment types that Amazon SES does not accept, see Unsupported attachment types (p. 513).

**Header fields**

Amazon SES can accept all email headers that follow the format described in RFC 822.

The following fields can't appear more than once in the header section of a message:

- Accept-Language
- acceptLanguage

**Note**

This field is non-standard. If possible, you should use the Accept-Language header instead.

- Archived-At
- Auto-Submitted
- Bounces-to
- Comments
- Content-Alternative
- Content-Base
- Content-Class
- Content-Description
- Content-Disposition
- Content-Duration
- Content-ID
- Content-Language
- Content-Length
- Content-Location
- Content-MD5
- Content-Transfer-Encoding
- Content-Type
- Date

**Note**

If you specify a Date header, Amazon SES overrides it with a timestamp that corresponds to the date and time in the UTC time zone when Amazon SES accepted the message.
- Delivered-To
- Disposition-Notification-Options
- Disposition-Notification-To
- DKIM-Signature
- DomainKey-Signature
- Errors-To
- From
- Importance
- In-Reply-To
- Keywords
- List-Archive
- List-Help
- List-Id
- List-Owner
- List-Post
- List-Subscribe
- List-Unsubscribe
- Message-Context
- Message-ID

**Note**
If you provide a Message-ID header, Amazon SES overrides the header with its own value.

- MIME-Version
- Organization
- Original-From
- Original-Message-ID
- Original-Recipient
- Original-Subject
- Precedence
- Priority
- References
- Reply-To
- Return-Path

**Note**
If you specify a Return-Path header, Amazon SES sends bounce and complaint notifications to the address that you specified. However, the message that your recipients receive contains a different value for the Return-Path header.

- Return-Receipt-To
- Sender
- Solicitation
- Sensitivity
- Subject
- Thread-Index
- Thread-Topic
- User-Agent
- VBR-Info
Unsupported attachment types

You can send messages with attachments through Amazon SES by using the Multipurpose Internet Mail Extensions (MIME) standard. Amazon SES accepts all file attachment types except for attachments with the file extensions in the following list.

**Note**
Some ISPs have further restrictions (such as restrictions regarding archived attachments), so we recommend testing your email sending through major ISPs before you send your production email.

<table>
<thead>
<tr>
<th>.ade</th>
<th>.hta</th>
<th>.mau</th>
<th>.mst</th>
<th>.psc1</th>
</tr>
</thead>
<tbody>
<tr>
<td>.adp</td>
<td>.inf</td>
<td>.mav</td>
<td>.ops</td>
<td>.psc2</td>
</tr>
<tr>
<td>.app</td>
<td>.ins</td>
<td>.maw</td>
<td>.pcd</td>
<td>.tmp</td>
</tr>
<tr>
<td>.asp</td>
<td>.isp</td>
<td>.mda</td>
<td>.pif</td>
<td>.url</td>
</tr>
<tr>
<td>.bas</td>
<td>.its</td>
<td>.mdb</td>
<td>.plg</td>
<td>.vb</td>
</tr>
<tr>
<td>.bat</td>
<td>.js</td>
<td>.mde</td>
<td>.prf</td>
<td>.vbe</td>
</tr>
<tr>
<td>.cer</td>
<td>.jse</td>
<td>.mdt</td>
<td>.prg</td>
<td>.vbs</td>
</tr>
<tr>
<td>.chm</td>
<td>.ksh</td>
<td>.mdw</td>
<td>.reg</td>
<td>.vps</td>
</tr>
<tr>
<td>.cmd</td>
<td>.lib</td>
<td>.mdz</td>
<td>.scf</td>
<td>.vsmacros</td>
</tr>
<tr>
<td>.com</td>
<td>.lnk</td>
<td>.msc</td>
<td>.scr</td>
<td>.vss</td>
</tr>
<tr>
<td>.cpl</td>
<td>.mad</td>
<td>.msh</td>
<td>.sct</td>
<td>.vst</td>
</tr>
<tr>
<td>.crt</td>
<td>.maf</td>
<td>.msh1</td>
<td>.shb</td>
<td>.vsw</td>
</tr>
<tr>
<td>.csh</td>
<td>.mag</td>
<td>.msh2</td>
<td>.shs</td>
<td>.vxd</td>
</tr>
<tr>
<td>.der</td>
<td>.mam</td>
<td>.mshxml</td>
<td>.sys</td>
<td>.ws</td>
</tr>
<tr>
<td>.exe</td>
<td>.maq</td>
<td>.msh1xml</td>
<td>.ps1</td>
<td>.wsc</td>
</tr>
<tr>
<td>.fxp</td>
<td>.mar</td>
<td>.msh2xml</td>
<td>.ps1xml</td>
<td>.wsf</td>
</tr>
<tr>
<td>.gadget</td>
<td>.mas</td>
<td>.msi</td>
<td>.ps2</td>
<td>.wsh</td>
</tr>
<tr>
<td>.hlp</td>
<td>.mat</td>
<td>.msp</td>
<td>.ps2xml</td>
<td>.xnk</td>
</tr>
</tbody>
</table>
# Amazon SES Developer Guide

document history

The following table lists the major changes to the *Amazon Simple Email Service (Amazon SES) Developer Guide*.

<table>
<thead>
<tr>
<th>Change</th>
<th>Description</th>
<th>Date Changed</th>
</tr>
</thead>
<tbody>
<tr>
<td>New feature</td>
<td>You can now specify a default configuration set (p. 252) to use when sending mail through each of your verified identities.</td>
<td>January 27, 2021</td>
</tr>
<tr>
<td>New feature</td>
<td>You can now receive notifications (p. 288) when an event occurs that causes the delivery of an email to be temporarily delayed.</td>
<td>June 19, 2020</td>
</tr>
<tr>
<td>New feature</td>
<td>Amazon SES is now available in AWS GovCloud (US) (p. 457).</td>
<td>April 30, 2020</td>
</tr>
<tr>
<td>New feature</td>
<td>You can now create Amazon SES endpoints in Amazon Virtual Private Cloud (p. 81) (Amazon VPC).</td>
<td>April 29, 2020</td>
</tr>
<tr>
<td>New feature</td>
<td>Amazon SES is now available in three additional AWS Regions (p. 457): Canada (Central), Europe (London), and South America (São Paulo).</td>
<td>April 1, 2020</td>
</tr>
<tr>
<td>New feature</td>
<td>You can now use your own IP ranges to send email. For more information, see Using Your Own IP Addresses to Send Email Using Amazon SES (p. 180).</td>
<td>December 23, 2019</td>
</tr>
<tr>
<td>New feature</td>
<td>You can now use your own public-private key pair to complete the DKIM authentication process for a domain. For more information, see Provide Your Own DKIM Authentication Token in Amazon SES (p. 138).</td>
<td>December 13, 2019</td>
</tr>
<tr>
<td>New feature</td>
<td>You can now use an account-level suppression list (p. 184) to automatically prevent sending messages to email addresses that previously resulted in a bounce or complaint.</td>
<td>November 25, 2019</td>
</tr>
<tr>
<td>New feature</td>
<td>If your account is in good standing, and you're approaching the sending quotas for your account, Amazon SES will automatically increase your quotas. For more information, see Increasing your Amazon SES sending quotas (p. 146).</td>
<td></td>
</tr>
<tr>
<td>Documentation update</td>
<td>Added information about deleting personal data from Amazon SES (p. 483).</td>
<td>March 13, 2018</td>
</tr>
<tr>
<td>Open sourced documentation</td>
<td>The Amazon SES documentation is now available on GitHub. You can submit issues or request changes in the GitHub repository, or make changes directly and submit a pull request.</td>
<td>February 22, 2018</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Added a section that provides information about deleting personal data (p. 483) stored in Amazon SES.</td>
<td>February 28, 2018</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Revised the Amazon SNS event publishing field definitions (p. 321), and added a Rendering Failure event example (p. 340).</td>
<td>January 22, 2018</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Date Changed</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Updated Deliverability Dashboard appendix to account for changes to IAM and Lambda consoles.</td>
<td>January 18, 2018</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> We removed this appendix on May 3, 2019, because it used components that were no longer supported.</td>
<td></td>
</tr>
<tr>
<td>Documentation update</td>
<td>Updated content related to publishing events to CloudWatch (p. 291) to mention blocked fields.</td>
<td>January 15, 2018</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Updated procedures for sending email using OpenSSL (p. 103) to make them easier to follow.</td>
<td>January 11, 2018</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Added code example for sending raw email by using the AWS SDK for Ruby.</td>
<td>January 2, 2018</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Added code example for sending raw email by using the AWS SDK for PHP.</td>
<td>December 29, 2017</td>
</tr>
<tr>
<td>New feature</td>
<td>Added content related to custom verification emails.</td>
<td>December 7, 2017</td>
</tr>
<tr>
<td>New feature</td>
<td>Added content related to pausing email sending and exporting reputation metrics for configuration sets.</td>
<td>November 15, 2017</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Added code example for sending raw email by using the AWS SDK for Java.</td>
<td>October 23, 2017</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Added code example for sending raw email by using the AWS SDK for Python (Boto).</td>
<td>October 20, 2017</td>
</tr>
<tr>
<td>New feature</td>
<td>Added content related to the email templates and personalized email features.</td>
<td>October 11, 2017</td>
</tr>
<tr>
<td>New feature</td>
<td>Added content related to the open and click custom domain feature.</td>
<td>September 18, 2017</td>
</tr>
<tr>
<td>New feature</td>
<td>Added content related to the reputation dashboard.</td>
<td>August 24, 2017</td>
</tr>
<tr>
<td>New feature</td>
<td>Added content related to dedicated IP pools feature.</td>
<td>August 17, 2017</td>
</tr>
<tr>
<td>New feature</td>
<td>Added content related to open and click tracking feature.</td>
<td>August 1, 2017</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Added an index of code examples.</td>
<td>June 26, 2017</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Added an appendix that demonstrates the process of creating a deliverability dashboard for Amazon SES.</td>
<td>June 22, 2017</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> We removed this appendix on May 3, 2019, because it used components that were no longer supported.</td>
<td></td>
</tr>
<tr>
<td>Documentation update</td>
<td>Updated email sending code examples.</td>
<td>June 6, 2017</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Date Changed</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for dedicated IPs.</td>
<td>November 21, 2016</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for email sending event publishing.</td>
<td>November 2, 2016</td>
</tr>
<tr>
<td>Service update</td>
<td>Updated to reflect that users no longer need to explicitly enable Easy DKIM signing after generating their DKIM records.</td>
<td>September 15, 2016</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Added a getting started tutorial for receiving email.</td>
<td>July 12, 2016</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for enhanced notifications.</td>
<td>June 14, 2016</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for custom MAIL FROM domains.</td>
<td>March 14, 2016</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for inbound email.</td>
<td>September 28, 2015</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for sending authorization.</td>
<td>July 8, 2015</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for AWS CloudTrail logging.</td>
<td>May 7, 2015</td>
</tr>
<tr>
<td>Service update</td>
<td>Updated to reflect the consolidation of the Amazon SES quotas increase forms and removed &quot;production access&quot; terminology.</td>
<td>April 8, 2015</td>
</tr>
<tr>
<td>Service update</td>
<td>Updated with new requirements for domain verification TXT records.</td>
<td>February 25, 2015</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Added Enforcement FAQ.</td>
<td>December 15, 2014</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for delivery notifications.</td>
<td>June 23, 2014</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for subdomain support.</td>
<td>March 19, 2014</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for Amazon SES expansion to the US West (Oregon) region.</td>
<td>January 29, 2014</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for Amazon SES expansion to the Europe (Ireland) region.</td>
<td>January 15, 2014</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated to reflect the changes in validation of Header Fields and MIME Types.</td>
<td>November 6, 2013</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Removed content on Sender ID.</td>
<td>August 22, 2013</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated to reflect the Amazon SES console redesign.</td>
<td>June 19, 2013</td>
</tr>
<tr>
<td>New feature</td>
<td>Renamed the blacklist feature to global suppression list.</td>
<td>May 8, 2013</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for the global suppression list removal feature.</td>
<td>March 4, 2013</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Added MIME types.</td>
<td>February 4, 2013</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Date Changed</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Included a Getting Started section to replace the stand-alone Getting Started guide, restructured the Table of Contents, and updated the Sendmail integration instructions.</td>
<td>January 21, 2013</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Added troubleshooting sections on increasing throughput and SMTP issues.</td>
<td>December 12, 2012</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Restructured the information on sending quotas.</td>
<td>November 9, 2012</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for the Amazon SES mailbox simulator.</td>
<td>October 3, 2012</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for using a DKIM signature to sign email from a verified identity.</td>
<td>July 17, 2012</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for receiving bounce and complaint feedback notifications through Amazon Simple Notification Service (Amazon SNS).</td>
<td>June 26, 2012</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for domain verification.</td>
<td>May 15, 2012</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated to reflect additional header and attachment types.</td>
<td>April 25, 2012</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for the STARTTLS extension to SMTP.</td>
<td>March 7, 2012</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for SMTP support.</td>
<td>December 13, 2011</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for AWS Management Console support.</td>
<td>November 17, 2011</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated for attachment support.</td>
<td>July 18, 2011</td>
</tr>
<tr>
<td>Initial release</td>
<td>This is the first release of the <em>Amazon Simple Email Service Developer Guide</em>.</td>
<td>January 25, 2011</td>
</tr>
</tbody>
</table>